Universidade do Porto

Faculdade de Engenharia

FEUP

Indicadores de Desempenho em Equipas de Desenvolvimento de Software

Mestrado Integrado em Engenharia Informática e Computação

Autor: João Pedro Pacheco de Figueiredo
Orientador: Nuno Flores
Instituição: Empresa do Sector Informático

26 de Janeiro de 2018

Indicadores de Desempenho em Equipas de Desenvolvimento de Software

João Pedro Pacheco de Figueiredo

Mestrado Integrado em Engenharia Informática e Computação

Abstract

Nowadays, software development projects are getting gradually more based on dynamic methodologies which promote process and product control and monitoring, in a way that the productivity of the ones who integrate them reaches the highest point among the necessities and goals of the given company.

Therefore, the dissertation 'Key Performance Indicators in Software Development Teams' arises from the partnership between Faculdade de Engenharia da Universidade do Porto and an IT Sector company, and it is fed by the necessity of that same company (leader in innovation and technology in numerous sectors) of implementing measurable metrics of execution, behaviour and performance relatively to its Software Development teams.

In this thesis, you will find elucidated the different types of methodologies adopted by the company's ongoing projects: the agile methods (iterative process that seeks rapid adaptation to changes in reality); waterfall (linear and sequential approach that starts with thoroughly settled requirements); and hybrids (the fusion of the two other methods to better suit the peculiarities of a certain project).

Consequently, development teams, depending on the followed methodologies, were evaluated according to the type of project in which they are, and later, carefully improved under software metrics and key performance indicators and introduced the aim of continuously measuring, improving their performance. Depending on the frequency of each indicator's measuring, one obtained quantitative results which enabled the comparison of reference values, and subsequently, the prominence of the areas in which the performance improvement should fall upon.

To point out that, parallel to the extraction of the results coming from the indicators, it was created an historical data bank, for continuous consultation and analysis with the intention of basing the estimation of future projects on those given results.

Concluding, this thesis tried to: (A) – Assess, process-wise, the given company's developing teams; (B) – Define a group of metrics and trustworthy key performance indicators in order to continuously enhance quality, productivity and efficiency; (C) – Implement these indicators in ongoing projects within the company; and, lastly (D) – extract results and, in case of an increase on the efficiency and quality of the areas of interest, as future work, it should be implemented an automated system of metrics and key performance indicators measurability.

Resumo

Atualmente, cada vez mais os projetos de desenvolvimento de *software* são baseados em metodologias dinâmicas que promovem a monitorização e controlo de processo e produto, de modo a que a produtividade de quem os integra atinja o desempenho máximo dentro das necessidades e objetivos de empresa em questão.

Como tal, a dissertação "Indicadores de Desempenho em Equipas de Desenvolvimento de Software" nasce da parceria instituída entre a Faculdade de Engenharia do Porto e uma Empresa do Sector Informático e é justificada pela necessidade que a mesma, líder em inovação e tecnologia nos mais diversificados sectores, sente em implementar métricas mensuráveis de execução, comportamento e desempenho relativamente às suas equipas de desenvolvimento de software.

Na presente tese elucidaram-se os diferentes tipos de metodologias adotadas pelos projetos em execução na empresa, tais como os métodos ágeis (processo iterativo que procura a adaptação rápida a mudanças da realidade), cascata (abordagem linear e sequencial, que parte de requisitos bem definidos) e híbridos (que aproveitam a combinação dos dois métodos anteriores para colmatar particularidades do projeto).

Por conseguinte, as equipas de desenvolvimento, dependendo das metodologias que seguem, foram avaliadas em conformidade com as tipologias do projeto em que estão inseridas, e, posteriormente, devidamente validadas segundo métricas de *software* e indicadores-chave de desempenho, introduzidos o intuito de mensurar, melhorar o seu desempenho continuamente. Dependendo da frequência das medições de cada indicador, foram obtidos resultados quantitativos que permitiram a comparação a valores de referência , e subsequentemente, salientar as áreas em que devem incidir as melhorias no desempenho.

De salientar ainda que, paralelamente à extração dos resultados provenientes dos indicadores, foi sendo criado um banco de dados históricos, para consulta e análise contínua com o intuito de que a estimação de projetos futuros possa alicerçar-se sobre esses mesmos valores.

Em suma, esta tese tencionou contribuir para: (A) - avaliar, a nível de processos, as equipas de desenvolvimento da empresa em questão; (B) - definir um grupo de métricas e indicadores-chave de desempenho confiáveis para efeitos de aumento contínuo de qualidade, produtividade e eficiência (C) - implementar estes indicadores a projetos que decorrem na empresa e, por último, (D) - extrair resultados e caso se verifique um aumento da eficiência e qualidade nas áreas de interesse, como trabalho futuro, deverá proceder-se à automatização da mensurabilidade das métricas e indicadores-chave de desempenho.

Agradecimentos

Antes de apresentar o documento, devo agradecer a todos os que me motivaram desde o início do meu percurso académico, que culmina com a escrita da presente dissertação.

Aos meus pais, pelo esforço emocional que toda a vida exerceram, colocando sempre em primeiro lugar, a minha educação, os meus valores e o meu bem-estar.

Ao meu avô por ser sempre um exemplo a seguir, pelos princípios de profissionalismo, entre-ajuda e perseverança que me foi transmitindo. À minha querida irmã pela motivação, carinho e competitividade, e ao resto da minha família pelo apoio incondicional nesta fase e durante toda a minha vida. Guardo um agradecimento especial à minha avó, dedicando-lhe mais esta conquista, por me ter criado e tornado na pessoa que sou.

À Helena, por ser o meu abrigo, pelo carinho, pela força e amparo em momentos que não seriam os melhores. Ela é parte integrante daquilo que eu sou.

Ao meu amigo Gonçalo Malafaya, meu companheiro desde os primeiros dias que me lembro do conceito de amizade, por acreditar nas minhas capacidades e me acompanhar de forma incansável ao longo destes anos. Se cheguei aqui, muito se deve a ele. Ao João Brito, por ser um pilar na minha vida, pelo apoio e pela capacidade de superação que sempre me incumbiu. Ao Nelson Novais pela paciência, espírito crítico e sentido de oportunidade que me ensinou durante esta fase. Aos meus amigos, pelo suporte que me deram e pelos momentos de desconcentração. Pelo facto de me acompanharem sempre de forma positiva em todos as fases da minha vida.

Agradeço ao Prof. Nuno Flores, pelo exemplo como pessoa e docente ao longo dos meus anos de estudo, pela disponibilidade e apoio durante a orientação deste trabalho de dissertação.

Em último plano, resta-me ressalvar a importância de todos os outros que me acompanharam e rodearam tiveram, ao longo de todos os passos estruturais da minha vida académica.

"Live as if you were to die tomorrow. Learn as if you were to live forever."

Mohandas Gandhi

Índice

Lista de figuras

Lista de Tabelas

Lista de Abreviaturas

IT	Information Technology
KPI	Key Performance Indicator
KRI	Key Result Indicator
ASD	Agile Software Development
BSC	Balanced Scorecard
CEO	Chief Executive Officer
RSD	Rapid Software Development
CC	Cumprir Calendário
TTC	Taxa de Tarefas Completas
CR	Custo de Reprogramar
CD	Custo de Desempenho
TF	Taxa de Falhas
SF	Satisfação de Clientes
ET	Entregue a tempo
PO	Product Owner
RSD	Rapid Software Development
ISACA	Information Systems Audit and Control Association
SEI	Software Engineering Institute
ISO	International Organization for Standardization
LoC	Lines of Code
KLoC	Thousand of lines of Code
US	User Story
COBIT	Control Objectives for Information and Related Technologies
CMMI	Capability Maturity Model Integration

1. Introdução

Nos dias que correm, o uso das tecnologias tornou-se imprescindível na maioria das funções do quotidiano, e quando se refere tecnologia, associa-se instantaneamente o termo *"software"*, que se caracteriza pelo comportamento de uma máquina segundo determinadas instruções escritas por programadores.

Com efeito, a produção de *software* é um ramo da indústria que tem crescido exponencialmente bem como o número das organizações que o realizam, muitas das vezes distintas pela qualidade do seu processo e pela adesão por parte dos seus clientes aos seus produtos. Assim sendo, existem fatores determinantes no que concerne ao desenvolvimento de *software* por parte de uma equipa de programadores, como a sua produtividade, fiabilidade, usabilidade, tempo de resposta e por fim a eficiência. O processo de produção do *software*, bem como a forma de o gerir, são fundamentais para que sejam cumpridos na sua plenitude os fatores que elevam e que colocam em destaque, o nome da empresa em questão.

Deste modo, as organizações que se dedicam à produção de *software* bem-sucedidas internacionalmente tendem a implementar cada vez mais, métodos assíduos de medição para controlo sistemático sobre o que é produzido a nível informático de todos os projetos que se encontram em desenvolvimento. Estes dados após extraídos e analisados, revelam informações objetivas que permitem efetuar mudanças atempadas afetando assim o desenvolvimento do produto e, consequentemente, o seu negócio, para um patamar superior.

Em suma, neste capítulo inicial pretende-se expor o contexto empresarial em que o estudo foi estruturado, a motivação que originou todo o interesse pelo próprio, os objetivos propostos e os resultados que se estimam obter, aquando da conclusão da mesma.

1.1 Contexto

A Empresa do Sector Informático sobre a qual se realiza esta dissertação é líder em inovação e tecnologia e visa solucionar as vicissitudes sociais, ambientais e económicas que nos rodeiam na contemporaneidade com projetos ambiciosos e visionários. Ao longo da escrita desta dissertação, por motivos de confidencialidade de dados, o nome da empresa em questão será substituído por **KPI Consulting**. Deste modo, a KPI Consulting é estruturada por sectores de IT ramificados em: equipas de desenvolvimento de software, de testes, de segurança e de consultoria, que em fase de conceção se revelam bastante cooperantes entre si.

Contudo, como todas as equipas que integram o desenvolvimento dos projetos de *software* revelam métodos muito próprios e exercidos ao longo dos anos da mesma forma, surge a necessidade de alguém exterior a estes departamentos avaliar os seus métodos. Sempre com o intuito de melhoria contínua a nível da gestão das equipas de desenvolvimento, recorre-se a métricas de diagnóstico de desempenho, implementando-as no decorrer dos projetos e, posteriormente, avaliam-se os seus resultados de modo a indicar quais serão os aspetos a melhorar. Assim sendo, ocorreram deslocações à empresa para extração de informação e interpretação dos procedimentos de trabalho, bem como foram realizadas reuniões semanais que visaram a recolha dos dados, consequentemente, a sua análise e discussão eventuais mudanças.

Um dos pontos fundamentais desta dissertação foi a integração em três projetos em desenvolvimento por parte da empresa do sector informático em questão, no sentido de os avaliar relativamente às suas metodologias tanto de código, como de gestão intrínseca das equipas. Segue-se um modelo de elucidação dos projetos que contém uma breve descrição dos mesmos, a equipa que o integra, a metodologia utilizada para o executar e algumas notas adicionais:

Projecto S

O projeto S trata-se de uma *framework* de gestão e integração de identidades da KPI Consulting. Cada um dos serviços que integram esta plataforma gerem as identidades fornecidas pelos Recursos Humanos de cada país no qual está presente a empresa. Dos processos envolvidos podem-se destacar a *SS-Tool* referente ao serviço de exportação dos dados para clientes da KPI Consulting, o MSS referente ao serviço de *mass-mailing*, entre muitos outros que gerem internamente o projeto na íntegra.

Equipa: O projeto é constituído por dois programadores locais, e dois remotos;

Metodologia utizada: O projeto S é desenvolvido em *Waterfall* [ver secção 2.1.1].

Notas adicionais: Um dos programadores remotos é o decisor do projeto, e suporta muitas das funcionalidades requeridas, o que pode acarretar consequências futuras (no caso da sua ausência);

Projeto P

O projeto P consiste no desenvolvimento de uma aplicação Web usada por unidades de negócio para pedir aconselhamento legal e fiscal sobre o registo de projetos internacionais. Como tal, é prioritário que a plataforma conglomere as propostas e adjudicações da KPI consulting, permita a definição de modelos de aconselhamento e de regras para a atribuição automática. Deve ainda possibilitar a comparação entre projetos para ajudar ao aconselhamento mais eficaz, suportar tanto a geração de relatórios, como a integração com outros sistemas para a criação automática e célere de novos projetos.

Equipa: O projeto é constituído por quatro programadores locais que se concentram na manutenção evolutiva da aplicação e prestam apoio à produção, e três colaboradores remotos que fazem a gestão do serviço;

Metodologias utilizadas: O projeto P é desenvolvido em *Waterfall* [ver secção 2.1.1], , ainda que, com a prospeção e análise para aderir aos métodos ágeis e faseado entre a recolha dos requisitos, desenvolvimento, testes de integração, testes de aceitação por parte dos utilizadores e só posteriormente a passagem a "live".

Notas adicionais: Devido a alteração dos requisitos, denotam-se atrasos significativos na fase de produção.

Projeto I

O projeto I denomina um sistema que permite ao setor dos recursos humanos gerir o processo de "delegação" a nível mundial. Existem actualmente 5 aplicações Web integrantes deste mesmo sistema: BUM (Business Unit Module), DCM (Delegation Center Module), DM (Delegation Module), WFM (Workflow Module) e WTM (Web Template Module). Todas elas são parte interveniente no conceito "delegação" da KPI Consulting e fornecem diferentes níveis de abstração e funcionalidades aos recursos humanos da empresa. Quando se fala em "delegação", fala-se de todos os processos que são necessários para a empresa prevenir quando um funcionário é destacado para um projeto/função num país diferente do seu país de origem. Muitas das vezes a necessidade de especialização leva a que um colaborador seja necessário num país estrangeiro e, assim sendo, é iniciado um processo de delegação para esse mesmo funcionário se encontrar enquadrado dentro legislação do país que o recebe, bem como obter todas as condições/regalias que foram acordadas para a sua deslocação (sendo ela de curta ou longa duração) e ainda garantir que, durante todo o tempo em que se encontra destacado, a KPI Consulting possui o seu processo atualizado e sincronizado com os outros sistemas de recursos humanos (como por exemplo o sistema de pagamentos e *reporting*).

Equipa: O projeto é constituído por seis programadores locais;

Metodologia utilizada: O projeto I foi o projeto pioneiro na tentativa de adoção de metodologias ágeis [ver secção 2.1.2]

Notas adicionais: A necessidade de comunicar continuamente durante o desenvolvimento do projeto, bem como a tentativa de monitorização das tarefas, levou a equipa do I, a adotar procedimentos mais ágeis.

1.2 Perceção de Qualidade no Processo

As equipas de desenvolvimento da KPI Consulting que integram os projetos descritos na secção 1.1 e sobre os quais vai incidir esta dissertação, possuem uma cultura de produção de software que se revela enraizada em métodos "tradicionais" como o *waterfall* (cascata). Ora, estas metodologias tendem cada vez mais a cair em desuso, sintoma comum a muitas organizações sobretudo de grande dimensão, daí surgir a necessidade de uma análise ao controlo da qualidade de produção de *software* praticado na empresa.

Deste modo, associada a uma procedimento conservador na produção irrompem problemáticas como: a indefinição no planeamento e na arquitetura do projeto, associada à estruturação ambígua dos requisitos do produto, na sua génese; a escassez de contacto entre a equipa de *developers* e o cliente, sintoma potencializado pela falta de redes viáveis de comunicação; a estimação significativamente superior do tempo de desenvolvimento, por forma a colmatar possíveis mudanças abruptas nos requisitos, surgimento de defeitos que levem à estagnação do processo, entre outras causas.

Por outro lado, após a seleção do processo de monitorização adequado, existe também um processo de adaptação dos métodos de desenvolvimento ditos clássicos para os mais ágeis ou híbridos. Prevê-se então, que uma das maiores adversidades esteja assente na mudança de ambiente de desenvolvimento pós migração, e a forma como irão lidar os membros das equipas com o seu novo mecanismo de desenvolvimento.

1.3 Objetivos e Motivação

Com efeito, o foco primordial desta dissertação é analisar as métricas [ver secção 2.3.1] e os indicadores chave de desempenho [ver secção 2.3.2] que mais se adequam à realidade das empresas de tecnologia e implementando os que, de facto, se adequam aos projetos da KPI Consulting. Ao mesmo tempo, dar-se-á uma tentativa de demonstração de metodologias mais ágeis dos processos de desenvolvimento, representam uma mais valia para a organização, em detrimento do mais usada até à data, o método em cascata.

Deste modo, aquando da comparação dos dados retirados de cada projeto antes e depois da aplicação dos KPI selecionados, espera-se obter valores de diagnóstico significativos, que permitam a definição de melhorias na eficiência dos processos, acrescentando por conseguinte, valor qualitativo e monetário á empresa.

No que concerne à motivação, para além da engenharia de *software* ser um ramo de particular interesse para carreira profissional, deve-se salientar que se trata de uma das maiores empresas a nível mundial, e, portanto, poder integrar e contribuir no aperfeiçoamento da mesma constitui a principal motivação.

1.4 Método seguido no projeto

O método para esta dissertação subdivide-se em cinco fases estruturais.

Primeiramente, a fase de reconhecimento empresarial com o objetivo de analisar a qualidade dos processos e do produto, bem como integrar os três projetos correntes na KPI Consulting.

Numa segunda fase, procede-se à pesquisa e definição dos indicadores chave de desempenho inerentes às empresas da área da tecnologia de informação. As questões de pesquisa baseiam-se na dificuldade de filtrar métricas e indicadores de desempenho específicos para a área de IT que melhoram a qualidade de software produzido no meio da variedade incomensurável de KPI que existem, e que, em paralelo, motivem a migração de métodos tradicionais para métodos mais ágeis. Assim sendo, recorre-se ao COBIT[12] (plataforma que reúne boas práticas e modelos de referência, criado pela ISACA, na gestão de empresas de tecnologia de informação) para a seleção de métricas mais usadas, bem como o exemplo da HP [13] na construção e gestão de um sistema de IT.

Posteriormente, em conjunto com a KPI Consulting e dependendo das necessidades dos projetos, dar-se-á a fase de filtragem dos KPI escolhidos [ver secção 4.2], restando apenas os que realmente serão implementados [ver secção 4.3].

No que concerne à quarta fase, esta resume-se especificamente na aplicação dos KPI filtrados, nos projetos que as tencionam integrar.

Finalmente, recorre-se à extração dos resultados, e encara-se esta fase como diagnóstico sobre a "saúde" do projeto. As fases mencionadas anteriormente são visíveis no plano de execução [figura 1.1] através de um gráfico de *Gantt*:

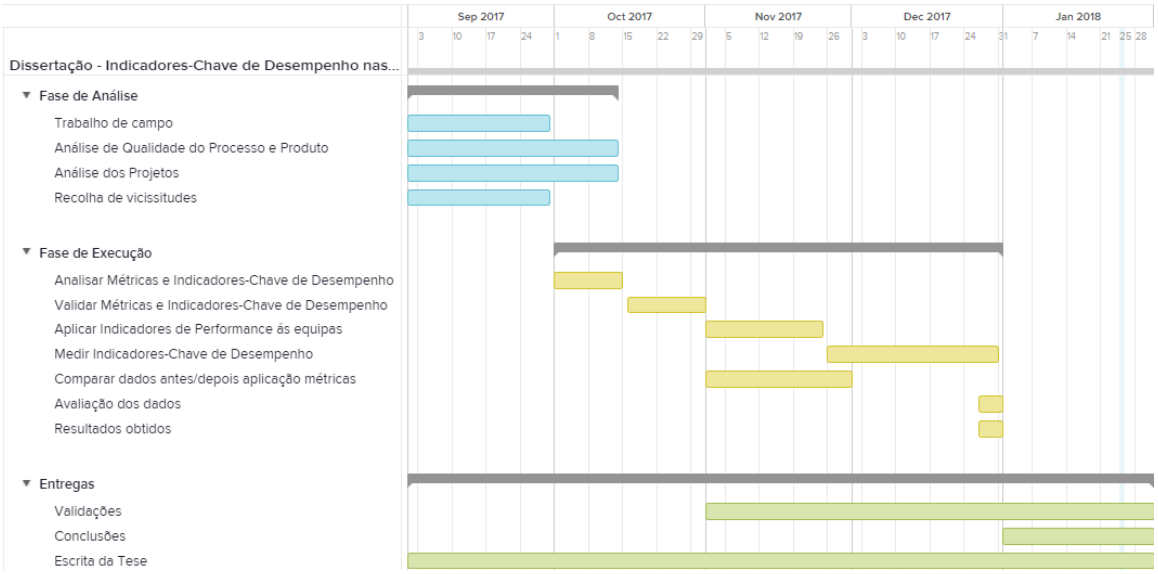

Figura 1.1: Gráfico de Gantt com o plano de execução de trabalho.

De salientar ainda, que as fases integrantes da solução e validação articulam-se no mesmo período, na medida em que constituem um processo iterativo e cíclico.

1.5 Resultados Esperados

De um modo sucinto, os resultados esperados elaboram-se através do cumprimento de todas as especificações propostas pela KPI Consulting nesta dissertação.

Primeiramente, o foco passou por pesquisar entre os que existem e definir indicadores-chave de desempenho relevantes de acordo com as tipologias dos projetos e os objetivos de suporte à melhoria contínua. Em segundo plano, validar com a organização a escolha anterior de forma a integrar os indicadores que especificamente se revelam uma mais valia a nível de monitorização e controlo da qualidade de processo e de produto. Segue-se a definição dos métodos de recolha dos dados, em conformidade com as equipas de desenvolvimento da KPI Consulting. Como quarta etapa, evidencia-se a identificação/obtenção de *baselines* e *benchmarks* que permitam estabelecer valores de referência para os indicadores-chave de desempenho destacados em fases anteriores, possivelmente dependentes de características dos projetos. E por último, salientar pontos fulcrais de melhoria nas áreas mais críticas com base nas métricas e valores obtidas de forma contínua mensalmente ou pelas diferentes entregas dos projetos.

Por conseguinte, através das fases descritas acima, pretendeu-se obter os seguintes resultados:

1. Lista de métricas de *software* que viabilizem a implementação dos indicadores-chave de desempenho;
2. Lista de indicadores-chave de desempenho que visem a melhoria do processo da empresa em questão;
3. Dados mensurados das diferentes áreas que caraterizam uma equipa de desenvolvimento nos projetos;
4. Dados analisados que permitam aferir as melhorias, ou o contrário, dos processos da empresa;
5. Estratégias concretas que validem a passagem de um método clássico para metodologias ágeis;

1.6 Estrutura

Além da introdução, este relatório estrutura-se em mais quatro capítulos.

O capítulo 2 serve de clarificação e enquadramento teórico dos conceitos e metodologias essenciais na realização desta proposta de dissertação.

Relativamente ao terceiro capítulo, o mesmo apresenta a situação inicial, expondo as questões em aberto evidenciadas pela realidade da empresa no período de de realização deste trabalho e pelos problemas detetados.

Por outro lado, o capítulo 4 refere-se a seleção dos KPI e à sua adoção por parte dos projetos da KPI Consulting, ou seja, numa abordagem mais de solução e contínua validação.

Por último, no capítulo 5 foi feito um balanço final de todo o trabalho realizado. Foram apresentadas as conclusões do projeto e foi analisado o sucesso de execução das soluções encontradas. Para finalizar foram identificadas algumas oportunidades de melhoria que poderão ser abordadas num futuro próximo.

2. Estado da Arte

Neste capítulo elucidam-se os conceitos teóricos de maior relevância que serviram de suporte para o desenvolvimento da presente dissertação.

2.1 Métodos de Desenvolvimento de Software

Hoje em dia, o processo de criação de *software* deixou de estar integralmente associado a código máquina, passando a exigir um método de trabalho específico e cauteloso. Entenda-se por método algo que está disposto numa ordem de progressão lógica e que visa atingir um objetivo.

Como tal, no ramo da tecnologia, existem três metodologias que se destacam no desenvolvimento de software, a linear, iterativa e a híbrida (que combina as duas anteriores). Neste caso em específico, abordaram-se dois métodos, não esquecendo que as equipas da KPI Consulting se guiam maioritariamente pelo modelo em cascata ou *waterfall*, e que a base desta dissertação está assente na tentativa de adoção de métodos mais ágeis.

2.1.1 Cascata ou Waterfall

O modelo em cascata ou *waterfall*, proposto por Royce (1970)[14], é denominado como modelo tradicional pela sua facilidade de compreensão e uso. Surge como uma abordagem sequencial e sistemática para o desenvolvimento de software, estruturado por várias camadas que só se iniciam após a conclusão integral da anterior, constituindo um fluxo uni-direcional para a frente (como numa cascata), sendo a principal dificuldade o retorno a fases anteriores.

Numa primeira instância, o modelo em cascata surge através do levantamento de requisitos do projeto, ou seja, são documentadas todas as necessidades e exigências por parte do cliente bem como as funcionalidades inerentes ao produto final.

Segue-se a fase de análise onde se definem prazos de entregas, componentes e modelos de negócio a usar para o cumprimento atempado do projeto.

Com o suporte e estudo das duas primeiras etapas, surge a arquitetura e design que representam a modelação estrutural do sistema, as suas tecnologias, seleção da linguagem de programação a utilizar, e os serviços que o mesmo requer.

Só na fase de implementação é que se dá início à escrita do código que garantirá, incrementalmente, a satisfação das necessidades outrora recolhidas.

Por último, na fase de testes, os seus intervenientes visam assegurar que as unidades desenvolvidas na fase de implementação não apresentam nenhuma falha, erro ou sejam danosas à entrega do produto final ao cliente, e que estão conforme exposto e requerido nos requisitos iniciais.

É de salientar ainda que, em alguns dos casos, pode existir ainda uma fase extra que representa a manutenção da plataforma para salvaguardar erros poste-

riormente encontrados. Todo este processo é descrito de forma intuitiva através da figura 2.1:

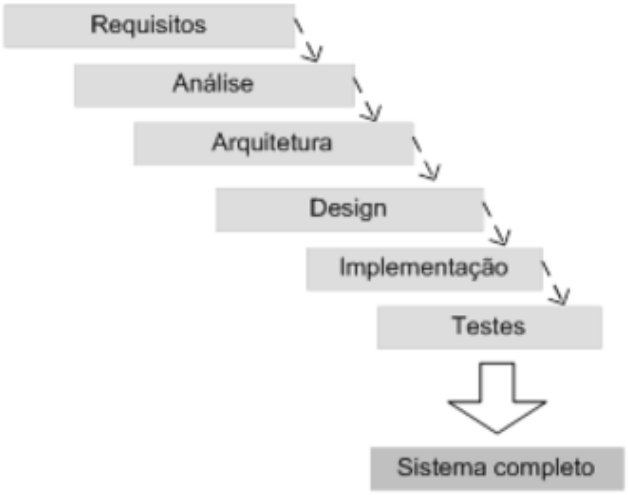

Figura 2.1: Método cascata de desenvolvimento de software. [1]

Em suma, o modelo visualmente elucidado pela figura 2.1, representa as fases estruturais do modelo em cascata, que espelha o seu objetivo na entrega faseada e sequencial do produto especificado.

Aplicação do método

Todos os projetos tecnológicos são diferentes, e no caso do método em cascata, devem-se ter em consideração alguns fatores internos (referentes à equipa de desenvolvimento que o vai estruturar) e externos (relativos ao cliente ou PO), para que seja apropriada a sua adoção.

Por conseguinte, são decisivos na adoção de um método em cascata fatores como: a clareza e transparência nos requisitos por parte do PO [ver secção 2.1.2], e a sua rigorosa documentação por parte dos membros integrantes do projeto; a definição do sistema é estável e não alvo de alterações estruturais durante a fase de implementação e a complexidade do projeto não deve ser elevada nem extensa a título temporal. [15]

Vantagens

- Fácil compreensão e uso;
- Permite alterações dos requisitos numa fase embrionária do projeto, na medida em que até à fase 5 (implementação), nenhum esforço foi feito a nível de código;
- Cada fase de desenvolvimento é precedida por outra em ordem estritamente incremental, o que concede o controlo de prazos para cada etapa;

- Como pressupõe uma documentação rigorosa, as equipas desenvolvimento depressa se adaptam a novos membros, ou à alocação de novas responsabilidades de outros;

Desvantagens

- Devido à estruturação sequencial no desenvolvimento, o contacto com o cliente prevê-se escasso após a primeira fase, tornando a mudança de requisitos tardia, um risco elevado para o projeto tanto monetário como temporal;
- Escassez de revisão para retroceder processos para fases anteriores;
- Modelo muito rudimentar no que concerne à complexidade dos projetos atuais;
- Fase de testes concebida unicamente após toda a implementação pode refletir-se na descoberta tardia de erros crassos que levem o projeto ao prejuízo;

2.1.2 Métodos Ágeis

Os métodos ágeis constituem uma alternativa sofisticada à gestão tradicional de projetos e começam a despontar o interesse das grandes empresas mundiais da área da tecnologia de informação. O termo *Agile* foi introduzido em 2001, por um grupo de entusiastas que visavam a entrega eficiente e de alta qualidade do produto final (RSD), como um movimento de oposição aos métodos clássicos. [16]

Por conseguinte, os métodos ágeis promovem um desenvolvimento adaptativo e sustentável que se adequa, como um pronto socorro, às imprevisibilidades irrompidas durante um projeto. Através de ciclos iterativos e entregas incrementais, o *Agile* permite receber, analisar e responder eficazmente a possíveis adaptações ou mudanças nos requisitos, a qualquer altura do desenvolvimento. As entregas incrementais surgem da fragmentação de um projeto complexo, em subprodutos mais simples, permitindo então que a equipa remodele a sua forma de agir ou pensar, salvaguardando danos maiores. Todo o processo está descrito visualmente através da figura 2.2:

É de extrema relevância referir que os métodos ágeis constituem não um caminho de mudança estrutural, mas sim de delineamento de preferências. Isto é, segundo o manifesto ágil[16], os mesmos promovem: (i) - os intervenientes do projeto e a sua interação contínua mais que processos e ferramentas na medida em que todos os projetos são constituídos por uma equipa de pessoas, e é fulcral o bom funcionamento entre elas para o levar a bom porto; (ii) - *software* funcional mais do que documentos; (iii) - colaboração contínua com o cliente mais do que somente o acordo do contrato inicial e a entrega final do produto; (iv) - adaptar a mudanças mais do que seguir um processo sequencial e estruturado. Assim sendo, um projeto deve ser planeado na sua génese, mas por outro lado, deve ser flexível o suficiente para albergar as mudanças repentinas que possam surgir.

Adicionalmente, outro dos avanços fulcrais dos métodos ágeis relativamente aos métodos clássicos, são as *sprint reviews*, ou reuniões regulares (semanal ou

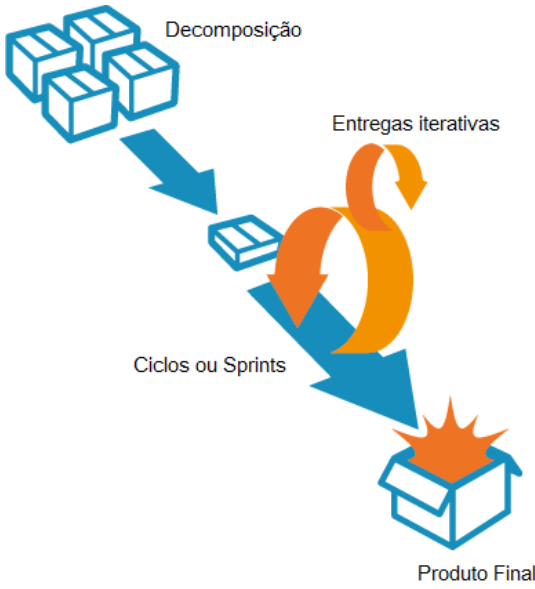

Figura 2.2: Métodos ágeis de desenvolvimento de software. [2]

mensalmente), que asseguram que as entregas sejam validadas pelo gestor do projeto, ou até mesmo pelo cliente como forma de controlo e aprovação de processos. Como tal, o cliente não precisa de esperar eternamente até à conclusão do produto para o analisar, muito pelo contrário, participa ativamente nas decisões que surgem, impedindo que o produto final se torne em algo que não corresponde ao esperado[17].

De seguida, enumeram-se alguns modelos ágeis de processo que se regem pelos princípios acima elucidados[18] :

- Scrum (1986) [19]
- XP (Extreme Programming - 1996) [20]
- Rapid Application Development (RAD - 1994) [21]
- Kanban [22]
- Feature Driven Development (FDD) [23]
- Dynamic Systems Development Method (DSDM) [24]
- Lean Development [25]
- Crystal [26]

Em suma, é de salientar ainda que, métodos como Scrum, XP, RAD, apesar de cada um com as suas particularidades, todos guiam o projeto visando um processo iterativo, eficaz na comunicação entre programadores e cliente, e baseado em respostas rápidas a alterações.

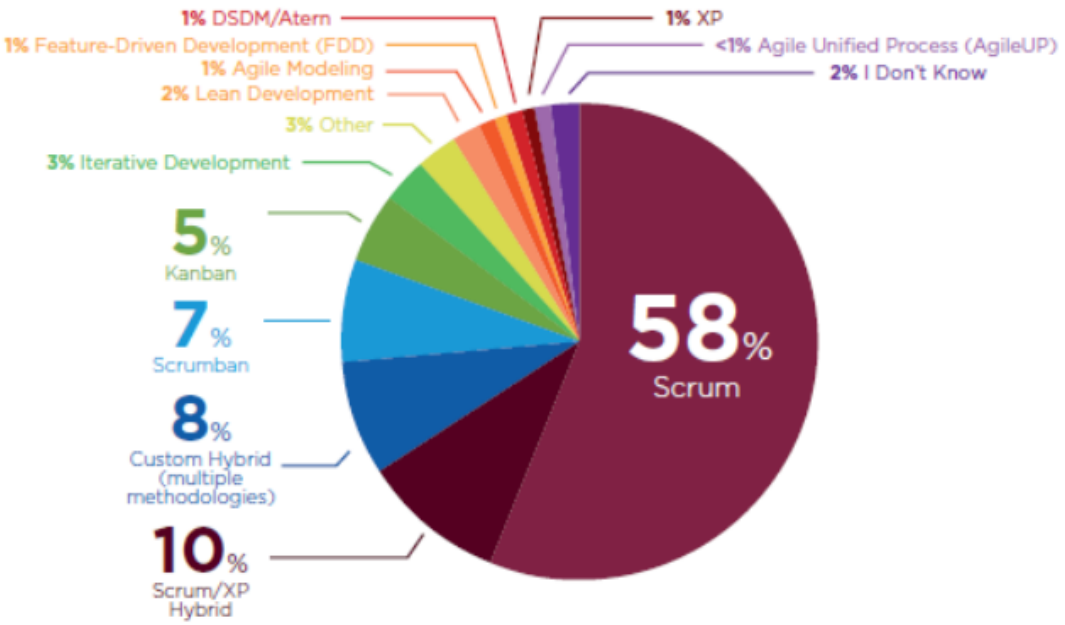

Figura 2.3: A percentagem da utilização do Scrum nas metodologias ágeis, relativamente a todos os outros modelos; (10th Annual State of Agile Report[3])

Aplicação do método

Para a aplicação de métodos ágeis como o *Scrum*, *XP* e afins num projeto, devem-se cumprir alguns requisitos para potenciar melhores resultados. Para além disso, a dificuldade de compreensão da ótica do *Agile* prende-se com o facto de se valorizar certas ferramentas e serviços em detrimento de outras (como no método em cascata, em que é estritamente necessário um planeamento exímio e bem documentado desde início).

Como tal, impõe-se a criação de vias de comunicação estáveis e eficazes, entre a equipa de desenvolvimento, o gestor do projeto e o cliente para uma constante atualização do estado dos processos.

Como se refere uma indústria dinâmica e repleta de atualizações, o plano inerente ao projeto deve ser flexível e adaptativo de modo a suportar mudanças a qualquer altura do desenvolvimento. Tanto a empresa como o cliente carecem de agendar com regularidade reuniões para a validação dos serviços até então conseguidos. [4]

Vantagens

- Satisfação do cliente pela entrega rápida, contínua e eficiente de amostras do software;[27]
- Alteração tardia nos requisitos não constitui um custo elevado (monetário ou temporal);
- Cooperação diária entre clientes e equipas de desenvolvimento de software; [27]

- Adequação dos serviços e tecnologias a utilizar, em prol da evolução sustentável do software;

Desvantagens

- Projetos extensos tornam árduo o esforço de cálculo necessário para o seu cumprimento atempado; [27]
- Pouco ênfase dado à documentação. modulação e arquitetura do projeto.

Scrum

Como observado através da figura 2.3, o *Scrum* é usado por mais de metade dos projetos que se guiam pelos métodos ágeis e dada a sua relevância no tema abordado e nas metodologias ágeis.

Assim sendo o *Scrum* é uma *framework* através da qual são propostos problemas complexos e adaptativos problemas, enquanto são desenvolvidas formas produtivas e criativas de os resolver e entregar o produto final segundo o mais alto rigor. De forma a otimizar a prevenção a riscos e falhas, controlar a produtividade dos seus intervenientes, o *Scrum* está assente sobre três pilares fundamentais. [28] Primeiramente, a transparência, na medida em que os aspetos comuns ao projeto devem ser mostrados e interpretados por todos os intervenientes; em segundo lugar, a inspeção, de forma a detetar variações indesejáveis no projeto, deve existir uma monitorização do que foi/está a ser feito contínua; finalmente, a adaptação, no caso de desvio de certos limites estabelecidos, ou funcionalidades que não se coadunam com o inicialmente especificado, uma vez que um ajuste deve ser tido em conta de imediato para minimização dos custos.

Segue-se uma listagem dos conceitos principais relativos ao uso da metodologia *Scrum* transversal a qualquer projeto, e a figura 2.4 que espelha os mesmos:[4]

1. **Sprint** - iterações que ocorrem durante todo o projeto, durante determinado tempo, e onde são implementas as funcionalidades relativas ao sistema a ser produzido;
2. **Product Backlog** - lista de requisitos funcionais e não-funcionais, que formulam o produto na sua íntegra, e o seu planeamento inicial de execução.
3. **Product Owner (PO)** - cliente do projeto, ou representante de vários clientes de um produto, e tem como principal responsabilidades a garantia de que os interesses, como cliente, são mantidos ao longo do projeto, bem como o cumprimento das funcionalidades de acordo com o estabelecido na lista de requisitos (*Product Backlog*);
4. **Equipas** - equipas que integram o desenvolvimento de software do projeto, assim como os seus testes, e têm as responsabilidade de implementar, a cada iteração, as funcionalidades especificadas;
5. **Scrum Master** - gestor do projeto, que tem como principal obrigação envolver todos os intervenientes na mentalidade *Scrum*, bem como obter e analisar os resultados, assegurando que todos seguem as regras e tipologias do método;

6. **Sprint Retrospective Meeting** - reunião no final de cada sprint, onde se verifica e é apresentado o que foi produzido pela equipa, e tem como objetivo a colaboração ativa de todos os intervenientes no projeto;

7. **Burndown Chart** - gráfico que mostra visualmente a quantidade de trabalho segundo o tempo que falta, apresentando um determinado ponto no tempo que, segundo o esforço medido, irá culminar no término do projeto.

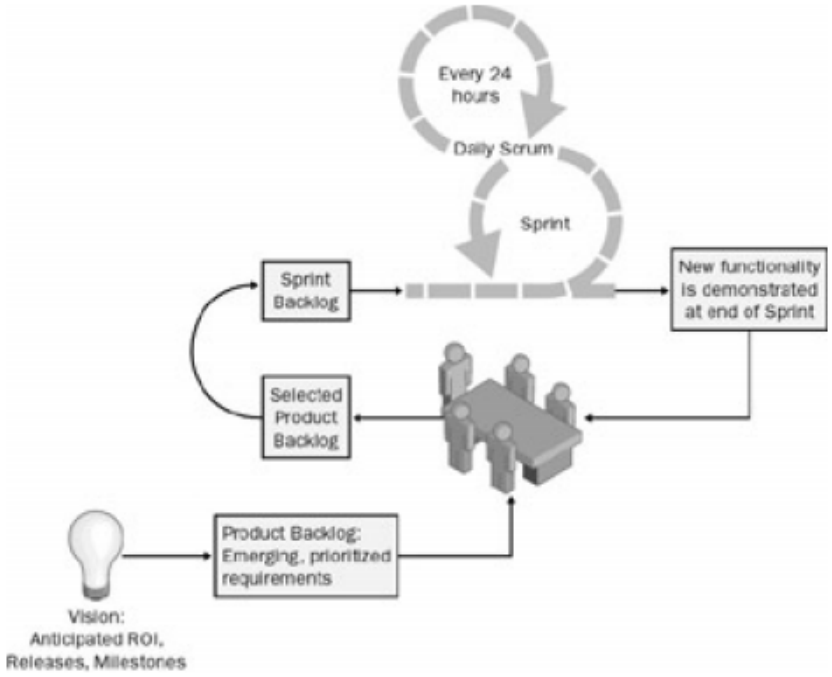

Figura 2.4: Processo *Scrum* e apresentação visual de conceitos como *sprint* e *product backlog.*[4])

2.2 Qualidade de Processo e de Produto de Software

Numa fábrica, o controlo de produtividade estima-se dividindo o número de peças produzidas pelo número de horas que cada colaborador trabalhou, porém, no sector informático, o caso muda de figura.

Para cada problema encontrado no *software*, surgem várias soluções possíveis, que podem ser mais eficientes por um lado, mais fáceis de manter por outro, mas onde no final, comparar as suas taxas de produtividade, pode tornar-se inconclusivo[5].

Assim, uma qualidade alta do processo de *software* está estritamente relacionada a um número irrisório de defeitos encontrados no mesmo, e se, de facto, cumpre com os padrões de eficiência, manutenção e segurança, por exemplo. Para estas garantias, é necessário que a organização implemente atividades de gestão de qualidade no processo e no produto, como a introdução de valores de referência [ver secção 2.3.3], planeamento qualitativo das funções nos intervenientes do projeto, e por último um controlo apertado na qualidade comparando o produto final

a padrões anteriormente definidos pela organização, ou por exemplos de outras da mesma indústria.

A referência principal na gestão da qualidade de uma empresa reflete-se no facto de que a qualidade dos seus procedimentos, vai afetar diretamente a qualidade dos produtos entregues.[5] Porém, no desenvolvimento de *software*, a relação entre a qualidade de processo e produto pode tornar-se complexa sem as formas corretas para as medir e indicar.

Veja-se a título de exemplo, um engenheiro de software pode escrever poucas linhas de código por hora, porém, estar a deixar código legado fácil de entender e, futuramente, com custos baixos de manutenção. Neste caso, deverá o gestor do projeto acusa-lo de não ser "produtivo"? Pois bem, o desenvolvimento de sistemas informáticos envolve criatividade e inovação e a experiência individual de cada afetará diretamente o produto final, ao contrário do que acontece com um produto mecânico, estandardizado para obtenção do mesmo resultado. Daí que a extração de valores e dados, no sector informático caso não exista um controlo de qualidade, facilmente pode tornar-se ambígua.

No que concerne ao controlo de qualidade de *software*, tem como base a monitorização regrada e contínua, de modo a que o projeto seja produzido em concordância com as especificações previamente definidas. Existem alguns modelos de referência que constituem as melhores práticas a nível de processo como os exemplos explicitados nas secções 2.2.1 e 2.2.2, o modelo CMMI e o ISO 9000.

Segundo Card[29], a receita para o sucesso no controlo de qualidade passa pela atribuição da responsabilidade a um agente externo ao desenvolvimento (gestor de projeto, por exemplo). É fulcral olear ferramentas automatizadas que facilitam a recolha das métricas e indicadores especificados, bem com a criação de um banco de dados comum, onde futuramente se retirem conclusões, comparações e valores de referência. Por último, a geração periódica de relatórios de dados, destacando ações a tomar e mudanças nos requisitos, sempre que preciso.

Por conseguinte, é de sublinhar ainda que o aperfeiçoamento da qualidade do processo leva à redução significativa de falhas na entrega do *software* [5]. Esta melhoria pode envolver processos como:

- Definir métricas de qualidade e indicadores chave de desempenho adequadas ao plano do projeto, bem como valores a serem atingidos como meta;
- Implementação e monitorização de métricas de qualidade e indicadores-chave de desempenho nos projetos durante o seu desenvolvimento, funcionando como garantia de controlo na produtividade;
- Reportar os resultados ao gestor do projeto e cliente, de modo a efetuarem uma análise detalhada dos resultados obtidos e serem tomadas medidas tanto relacionadas com o rumo do projeto, como sobre o futuro de quem o realizou.

A figura 2.5 pretende traduzir visualmente os procedimentos que envolvem o controlo da qualidade de processo e de produto, bem como as suas diferentes fases.

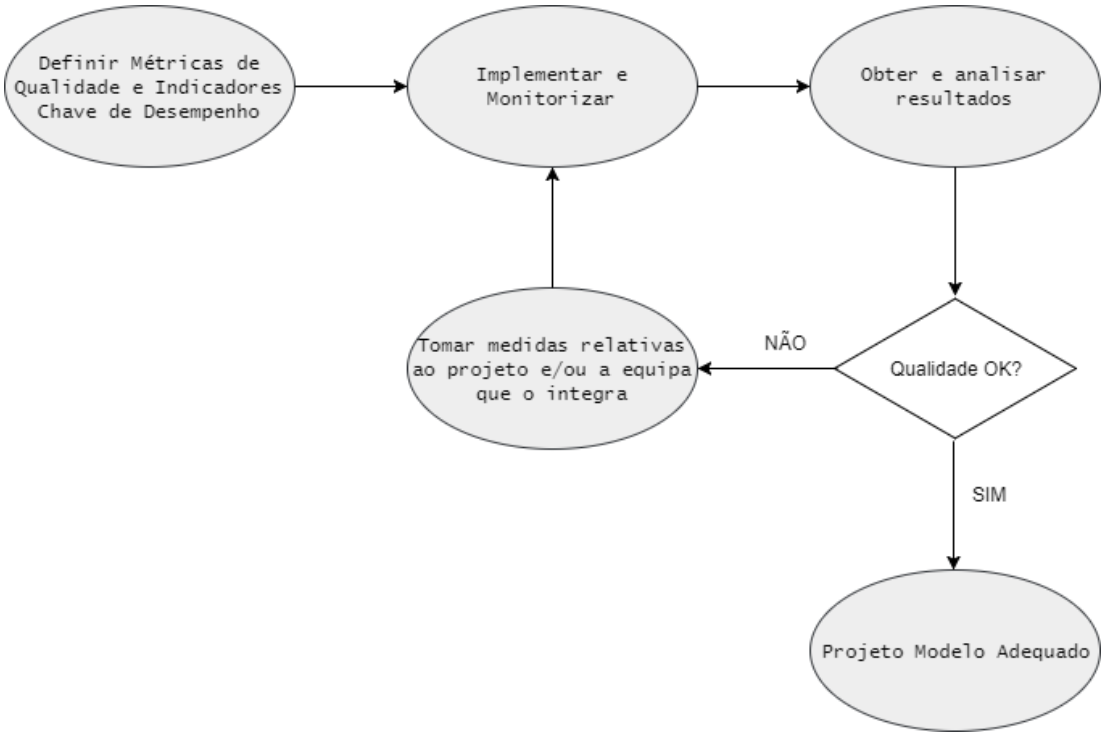

Figura 2.5: Processo de gestão de qualidade adaptado.[5]

Em suma, é fundamental que as organizações estimem a produtividade das suas equipas de desenvolvimento bem como a qualidade e custo dos seus processos através de métricas quantitativas e indicadores-chave de desempenho, conceitos aprofundados na próxima secção.

2.2.1 Modelo CMMI

O Capability Maturity Model Integration (CMMI), é um conjunto de valores de referência que padronizam as organizações que a ele recorrem de modo a estabelecer melhorias contínuas a nível da qualidade dos seus processos. [30]

Os modelos CMMI, promovem a melhoria na qualidade e eficiência dos processos das organizações, através da imposição de um ritmo eficiente no desenvolvimento de software. Para além disso, foca-se também da remoção da ambiguidade e inconsistência nos processos, no estabelecimento de padrões e regras uniformes para os setores abrangidos, no aumento da mentalidade partilhada, assegurando a promoção de um estilo de procedimentos adequados a todos por igual.

O CMMI recorre a níveis de processo de modo a categorizar o estado em que se encontra determinada organização ou uma determinada área da mesma. As empresas que se encontram na fase 1, executam diversos projetos, porém, a sua monitorização não ocorre, a comunicação entre os intervenientes é escassa, e não existe nenhum processo bem definido. Na fase dois, o processo começa a ser gerido, para na terceira fase representar um processo com a principal característica sendo a pro-atividade dentro da organização. O nível 4, reflete um processo que é devidamente mensurado e monitorizado quantitativamente em todas as suas áreas e, no final, a etapa 5 é estruturada unicamente pelo foco na melhoria contínua dos processos das empresa. [16]

Todas estas etapas são visualmente explicitadas através da figura 2.6.

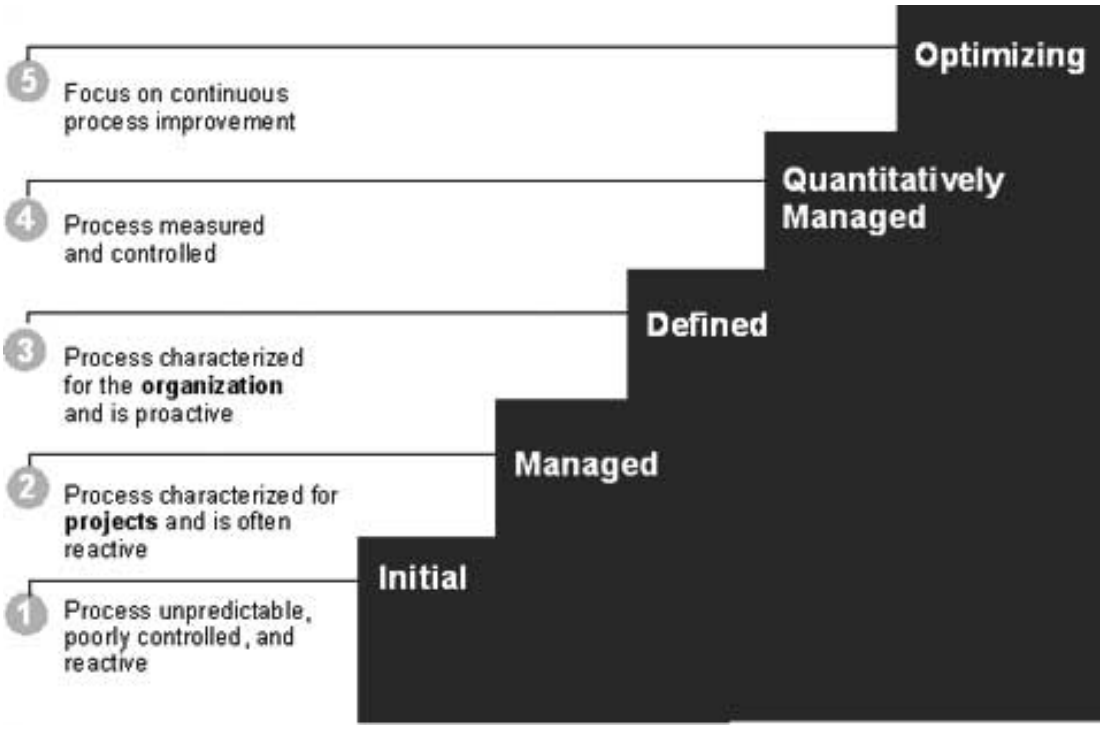

Figura 2.6: Níveis de maturidade do modelo CMMI.
[31]

Em suma, o modelo CMMI é explicitado através de níveis de maturidade organizacional, e, dependendo da organização e da sua capacidade de gerir os seus processos, o nível vai aumentando proporcionalmente à sua reputação.

2.2.2 Modelo ISO 9000

A *International Organisation for Standardisation* ou ISO, é um conjunto de organismos de vários países, que tem como principal responsabilidade, a normalização e estabelecimento de um consenso internacional sobre boas práticas de gestão.

As normas impostas pelo modelo ISO são revistas, de 5 em 5 anos, pela comissão responsável, com a missão de se manterem atualizadas, eficazes e adequadas. Como tal, as empresas necessitam de se atualizar de forma sistemática e recorrente para obter o último certificado lançado pelo modelo, uma vez que os anteriores são válidos por um período de tempo, após o qual se tornam obsoletos.[32]

No que concerne à gestão da qualidade dos sistemas numa organização, esta é suportada pelas normas referentes à série ISO 9000. *NP EN ISO 9000* - Sistemas de gestão da qualidade. Fundamentos e vocabulário, *NP EN ISO 9001* - Sistemas de gestão da qualidade. "Requisitos, *NP EN ISO 9004* - Sistemas de Gestão da Qualidade – Linhas de Orientação para Melhoria de Desempenho.

As diferentes séries destacam-se através das medidas que implementam, veja-se a título de exemplo os objetivos da implementação da ISO 9001: [32]

- Fornecer um contexto fidedigno e estável dos requisitos para os próximos

dez anos;

- Ser genérica, continuando ainda a ser relevante para todos os tipos e dimensões de organização, independentemente do tipo de indústria ou setor;
- Contemplar as recentes alterações da prática da gestão da qualidade, da tecnologia e do ambiente de trabalho, cada vez mais complexo e dinâmico, por forma a enquadrar a sua relevância prática;
- Manter o foco presente numa gestão eficaz dos processos;
- Simplificar a implementação nas organizações e a avaliação da conformidade;
- Simplificar o texto dos requisitos para garantir uma compreensão idêntica e interpretações uniformes;

Em suma, existem determinados modelos que alertam as empresas para a importância da qualidade do processo e do produto que produzem, com o intuito de maximizar a sua eficiência.

2.3 Medição no desenvolvimento de Software

Na engenharia de *software*, a medição no desenvolvimento, explicita-se na tradução numérica de um atributo ou processo, através da qual se efetuam comparações com valores de referência. As mesmas permitirão aferir conclusões acerca do estado do projeto e da sua qualidade. Por conseguinte, o uso desta medição visa obter previsões atempadas sobre o sistema a ser desenvolvido (por exemplo, caso existam demasiadas anomalias, o projeto provavelmente será mais custoso do que proveitoso). E, por outro lado, permite a identificação imediata de componentes cujas caraterísticas fogem do previsto e estipulado pela empresa. [33]

Veja-se a título de exemplo, uma organização que tenciona introduzir uma ferramenta nova de teste de *software*. Antes da sua introdução, recolhe-se o número de defeitos detetadas no sistema, durante um mês. Após a adoção da ferramenta, volta a repetir-se a medição da métrica - "Número de Falhas Funcionais", ao longo do mesmo tempo. Caso se verifique a deteção de mais falhas, no mesmo período de tempo desde a adoção, pode significar que o software clarifica o processo de validação e teste da empresa[5].

Assim sendo, a medição na produção de software, a sua análise e avaliação (processo visualmente explícito na figura 2.6), constituem um ferramenta poderosa na gestão das empresas do sector informático, e podem representar um avanço fundamental num ambiente competitivo que as circunda relativamente a outras. Citando Hughes, - *"O que for medido, será melhorado, o que não for, será ignorado."*[33]

Em suma, também o processo de medição no desenvolvimento de *software* é estruturado ciclicamente [ver figura 2.7], através de medições, da sua análise, dá-se a melhoria, e a sua eficiência é aumentada de forma contínua.

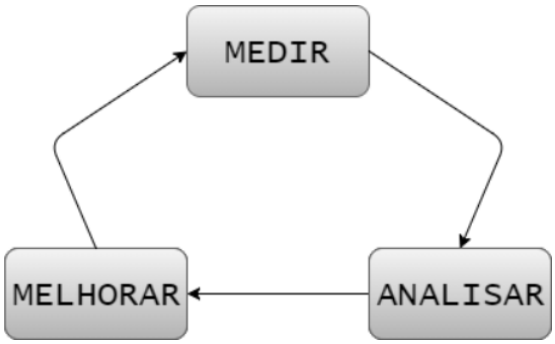

Figura 2.7: Ciclos do processo de medição no desenvolvimento de projetos de software.

2.3.1 Métricas de Qualidade de Software

Anteriormente, elucidaram-se as caraterísticas que distinguem um método tradicional e em desuso dos métodos ágeis, caraterísticas que reivindicam a estipulação de métricas de qualidade de software, indicadores chave de desempenho e um processo de medição adequado à sua abordagem [34].

O SEI (Software Engineering Institute), declara as métricas de qualidade de software, como um conjunto de medidas de um processo ou atributo de software, que estimam quantitativamente a produtividade do mesmo, o seu custo, a sua qualidade e a sua eficácia ao longo do seu desenvolvimento[35]. Como objetivo primordial, estas métricas identificam e medem os principais parâmetros que possam afetar o desenvolvimento do *software* empresarial, prevenindo de uma maneira organizada e detalhada, a falha na gestão de determinado projeto, que pode significar prejuízos avultados. Além do mais, a definição de um conjunto detalhado de métricas apoia a medição contínua do produto, mede a sua qualidade e dá suporte à tomada de decisão por parte do gestor do projeto, por exemplo, garantindo que o seu processo é permanentemente monitorizado, adaptado e melhorado. [36]

Ao longo dos últimos anos, uma quantidade elevada de métricas de *software* têm sido implementadas em projetos de naturezas diferentes, o que eleva o risco de falha aquando da sua escolha para o projeto requerido. De salientar o facto de que é primordial a justificação minuciosa na escolha das métricas, para evitar que o seu uso não evolua na direção contrária dos problemas.[37] A escolha inapropriada das métricas a utilizar pode levar à não conclusão ou estagnação do projeto, na medida em que potencia o esforço para o terminar, e pode orientar o gestor de projeto a tomar decisões inconclusivas ou equivocadas. Adicionalmente, a grande vicissitude da implementação exclusiva destas métricas (sem indicadores) prende-se com o facto de que são facilmente ludibriadas. Veja-se a título de exemplo, a métrica *"número de linhas de código"*, o colaborador pode, indubitavelmente, escrever o dobro das linhas de código, só com comentários e funções semelhantes replicadas. Assim sendo, seguem-se alguns requisitos para a escolha de métricas, que devem ser levados em conta [8]:

- O alvo da medição deve ser inequívoco e objetivo;
- Adaptativa nas mudanças de requisitos, caso existam;
- Difícil de manipular;

- Permitir uma contínua e consistente medição;
- A sua implementação deverá consumir um valor residual de recursos (esforço e monetário);
- Todas as medições e recolhas de dados devem ir de encontro aos objetivos da empresa e do projeto;

Benefícios do uso de Métricas de Software

Com a atribuição de valores de referência, pelo IEEE[38] ou ISO, a comparação e consequente análise dos dados recolhidos através de métricas de software tornou-se vital na grande maioria dos projetos. Por conseguinte, estima-se que a implementação prévia de métricas no software comporte os seguintes benefícios: (i) - estrutura princípios de qualidade para um sistema, desde a sua génese; (ii) - padroniza critérios facilitando por outro lado a sua validação; (iii) - prevê continuamente o custo do produto ao longo do todo o seu desenvolvimento; (iv) - comparando os resultados com valores de referência, avalia o nível de qualidade de forma consistente; (v) - possibilita a comparação do histórico de valores do projeto atual, com projetos outrora concebidos; (vi) - facilita uma estratégia de gestão pro-ativa estritamente relacionada com a deteção e prevenção de falhas no sistema.

Em jeito de conclusão, o principal benefício do uso de métricas de software passa pela identificação das áreas onde é prioritária uma melhoria imediata.

Exemplos de Métricas de Software

Os métodos ágeis dividem-se em diversas fases tais como a arquitetura do software, a sua implementação e a parte fundamental que são os testes ao sistema. Como tal, foram também criadas métricas que integram cada uma dessas fases, de modo a que todo o processo seja coberto por medições que geram uma melhoria contínua.

Arquitectura

- Número total de pontos de função;
- Tempo estimado para cada tarefa;
- Número de Requisitos;
- Número total de Ficheiros;

Implementação

- Número total de falhas no código;
- Número de linhas de código (LoC ou KLoC);
- Número de falhas corrigidas;
- Tempo gasto a corrigir erros;

Teste

- Número total de testes unitários;
- Severidade;
- Número de falhas encontradas;

Em suma, deve-se salientar ainda que os exemplos acima descritos possibilitam a medição indireta de atributos de qualidade de software, mais propriamente requisitos não-funcionais como: a manutenção do produto final, a sua usabilidade, portabilidade e fiabilidade. Veja-se pela figura[5]:

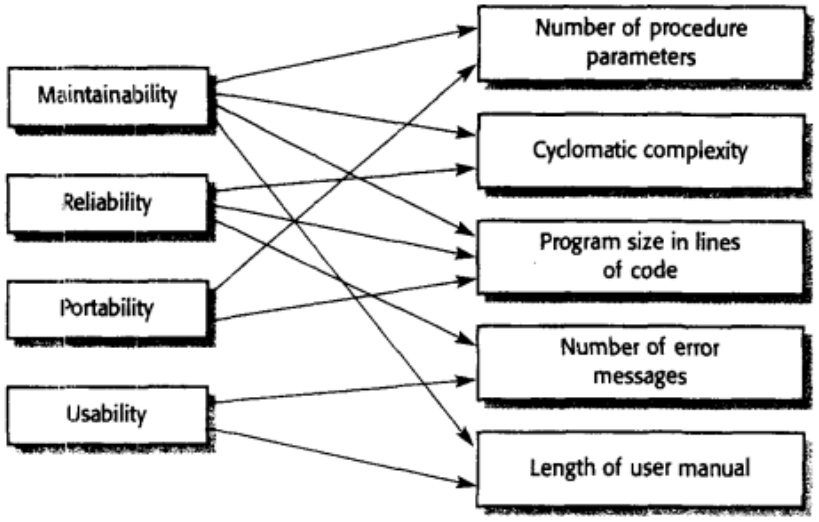

Figura 2.8: Relação entre os atributos externos como usabilidade, portabilidade, fiabilidade e as métricas que possam ser implementadas no projeto.[5]

2.3.2 Indicadores-Chave de Desempenho (KPI)

No que concerne ao desenvolvimento de software, os indicadores-chave de desempenho, do inglês *Key Performance Indicators* (em algumas secções abreviados para KPI), representam um conjunto de medições cujo principal foco é guiar e analisar aspetos de sucesso presentes e futuros do funcionamento de uma empresa. A par disso, os KPI possuem objetivos estratégicos bem definidos, e a sua medição frequente espelha-se no quão próxima estará a empresa de alcançar determinado propósito. De uma forma esclarecedora, na antecedente secção, elucidou-se que métricas são medidas quantificadas e numéricas sobre algum aspeto do projeto ou organização. E do outro lado, os KPI representam também métricas, ou um conjunto delas, tendo como objetivo o alerta e a indicação da distância que poderá estar a empresa-alvo da sua meta final conforme o medido. [39]

Segundo Kaskinen[40] - "... um KPI concede a uma empresa a possibilidade de medir o seu desempenho atual comparando-a a objetivos e medidas de referência, de forma a perceber as suas forças e onde é que está a falhar neste preciso momento."

Veja-se a título de exemplo, a British Airways (BA), que em 1980, se concentrou no uso de indicadores de desempenhos e na consequente entrega de relatórios semanais[41]. Um dos mais notáveis KPI implementado foi - "Número de aviões da BA atrasados por número de passageiros", pelo que, por cada aumento inesperado no valor deste indicador, o CEO da companhia aérea seria notificado, onde quer que estivesse no mundo. Medições diárias deste KPI, fizeram com que os colaboradores recebessem chamadas telefónicas do CEO a questionar o porquê dos atrasos, e como seria de esperar, a BA celeremente se tornou na transportadora aérea com uma reputação notável no que concerne a aviões chegados no tempo estipulado. A introdução deste KPI afetou positivamente a empresa em algumas perspetivas como: (i) reduziu custos como devoluções na íntegra de viagens aos passageiros e a sua consequente alocação em hotéis; (ii) aumentou a satisfação dos clientes; (iii) reduziu a emissão de gases poluentes para a atmosfera uma vez que o combustível adicional usado pelo avião para chegar ao seu destino a tempo por ter saído atrasado deixou de existir; (iV) providenciou um impacto direto na mentalidade de todos os funcionários que por cada avião atrasado, receberiam uma chamada direta do dono da companhia a pedir uma justificação, bem como teriam que lidar com a insatisfação dos clientes; (v) constituiu uma melhoria significativa do serviço prestado no geral[41].

David Parmenter, através de conferências e workshops referentes a indicadores de desempenho, consolidou alguns requisitos comuns à sua aplicabilidade tanto em empresas do sector privado como do público. Como tal, os KPI podem ser definidos por algumas das características que se seguem[41]:

- Devem ser claros, específicos e com um objetivo bem definido;
- Devem ser mensuráveis e comparáveis;
- Não se expressam através de medidas financeiras (não pode ser expresso em dólares, euros);
- Passíveis de validação estatística;
- Promovem ações de melhoria, não de levantamento de problemas maiores;
- São medidos regularmente (diariamente ou semanalmente) para garantir eficácia nos dados obtidos;
- São propostos e implementados por um agente externo, CEO ou gestor do projeto;
- Esclarecedores e profundamente compreendidos por toda a equipa de desenvolvimento;
- Agregam a responsabilidade a todos os intervenientes do projeto para o seu cumprimento;
- Provocam um enorme impacto nos núcleos críticos de sucesso e nos BSC;
- Afetam todas as medidas de desempenho de uma forma positiva;

Figura 2.9: Combinação entre Tecnologia e Negócio, resultando nos KPI's. [6]

Com efeito, os KPI são então, métricas fundamentais que medem de forma eficiente o progresso e o desempenho e requerem uma compreensão proficiente do que é importante para a organização e quais as suas metas.

Benefícios do uso de KPI

Desde a sua implementação, os KPI permitem à organização uma medida contínua do seu desempenho, o que os tornam numa ferramenta de melhoria contínua. Assim clarificado, os principais motivos da integração de KPI numa empresa de desenvolvimento de *software* são[41]:

- Dados facilmente recolhidos, guardados e passíveis de ser recuperados;
- Providenciam dados históricos para estudos posteriores sobre a empresa e seus métodos de desenvolvimento;
- Revelam as potenciais fraquezas, e por conseguinte determinam ações que necessitam ser implementadas, e em que núcleo surgem;
- Através da sua aplicabilidade as empresas podem observar no que se tornarão futuramente;
- Permitem identificar oportunidades de poupança nos custos do projeto e arquitetam fórmulas de controlar futuros gastos desnecessários;
- Criam uma atmosfera aberta de partilha comunicativa entre equipas, guiando-as para um objetivo comum e para a gestão de qualidade do software;

Exemplos de KPI

Existe uma variedade de KPI em todos os ramos de negócio empresarial, porém na presente dissertação, evidenciam-se aqueles que se correlacionam com o domínio do desenvolvimento de *software*.[42]

Por conseguinte, existem indicadores de **processo**, tais como a velocidade de produção e sprints da equipa a programar a plataforma inteira, e, por exemplo,

o tempo de espera entre as ordens do cliente de mudança ou incrementação de tarefas no projeto e a concretização das mesmas.

Por outro lado, subsistem as métricas de **produção**, nomeadamente o tempo médio entre falhas, que tal como o nome indica, espelha a frequência com que ocorrem falhas, o tempo médio entre reparação/recuperação e por fim, a taxa de falha da plataforma no culminar do processo.

Por último, outro exemplo, surge das métricas de **cliente** que se caraterizam pelo número de pedidos de alteração ou adição de componentes ao *software* a desenvolver, a sua satisfação, .

Processo de Introdução de KPI numa Empresa

O processo de introdução de KPI numa empresa, projeto ou equipa de desenvolvimento, pode ser personalizado visando colmatar procedimentos e necessidades estruturantes das organizações. Tal elucidado, este secciona-se em 5 fases estruturantes.

Primeiramente, a fase de identificação dos indicadores tendo em conta o contexto empresarial, os métodos de desenvolvimento inerente às equipas, bem como os problemas associados à adoção de uma nova ideologia de monitorização. Numa segunda etapa, dá-se a validação dos KPI anteriormente destacados, consistindo numa filtragem dos mesmos para uma adesão cautelosa aos processos da empresa ou do projeto. De seguida, a implementação dos indicadores no contexto para o qual foram definidos, implicando assim uma medição diária, semanal ou mensal das métricas necessárias para os preencherem. Numa quarta fase, analisam-se os resultados das medições realizadas no procedimento anterior, com o intuito de avaliar o progresso, ou não, do KPI segundo o seu objetivo previamente definido. Por último, e segundo as avaliação dos dados obtidos, dá-se a fase da tomada de decisão, por parte do CEO, agente externo ou gestor de projeto, para que o foco seja mantido e a empresa continue a prosperar.[41]

Figura 2.10: Etapas de introdução de KPI numa empresa.[**?**]

Deve-se salientar, que a implementação de um único KPI não permite uma visão geral sobre o desempenho das equipas. De tal modo que a gestão dos processos deve ser baseada em fatores determinantes contíguos ao alinhamento estratégico da empresa, e sempre visando os indicadores que terão mais impacto na realidade da empresa.[40]

Em suma, os indicadores-chave de desempenho permitem avaliar o estado de um projeto a decorrer, acompanhar os potenciais riscos associados, detetar

as áreas passíveis de melhoria, ajustar o fluxo e esforço de trabalho de cada interveniente, e avaliar a capacidade da equipa de produzir o produto especificado, bem como a sua qualidade. [43]

2.3.3 Valores de Referência (Benchmarking)

Os valores de referência ou *Benchmarking*, é um processo que compara os resultados obtidos num determinado contexto (neste caso dos KPI), com os valores apropriados obtidos pela indústria ou pelo seu setor, fazendo com que a empresa tenha sempre objetivos definidos na melhoria do seu desempenho.

Os valores de referência também constituem uma mais valia para as empresas na medida em que estabelecem uma linha de ação sobre a identificação de áreas mais necessitam de atenção após a comparação com outras. Deste modo, a organização consegue também aumentar a sua produtividade, qualidade e competitividade.

É de fulcral relevância salientar que, as comparações podem ser estabelecidas através de: (i) - medições de empresas similares; (ii) - internamente, entre diferentes setores e áreas da empresa em questão; (iii) - medições da empresa comparativamente a empresas do topo mas da indústria; (iv) - e por último, medições da empresa comparativamente a empresas de topo, independentemente da indústria em questão. [44]

A obtenção dos valores de referência também permitirá à empresa saber mais sobre a indústria e organizações do ramo que a rodeiam e assim, estabelecer metas de alcance às mesmas, redirecionar-se para o caminho correto, e até modificar padrões da empresa de modo a torná-la mais produtiva.

Existem quatro passos que estruturam o processo de *benchmarking*. Em primeiro plano, surge a análise à organização, aos seus processos, procedimentos e ao seu desempenho, que permitirá o ponto de partida para a definição das áreas a melhorar. De seguida, pesquisar sobre os valores de referência que se pretendem adotar, bem como a empresa à qual se quer assemelhar, constituindo o passo mais importante de todos, na medida em que a escolha desadequada pode levar a estratégias erradas e a leituras irrealistas de dados. O terceiro passo, envolve a extração de dados da empresa e dos procedimentos estipulados anteriormente, para consequente comparação. E, por último, a compilação de todas as informações recolhidas de modo a decidir sobre quais as estratégias e metas a adotar para a melhoria dos processos da empresa.[45]

Em suma, através das comparações proporcionadas pelos valores de referência, as empresas que introduzem este sistema visam tomar decisões a partir da informação analisada, ter uma noção mais abrangente sobre os riscos de determinadas áreas, bem como torná-la mais eficiente.

2.4 Sumário

Em síntese, no presente capítulo elucidaram-se os diferentes métodos de desenvolvimento de *software*, na secção 2.1, que contam com uma abordagem sobre os métodos clássicos como o caso do cascata, e de metodologias mais ágeis. Apresenta-se ainda a figura 2.11 que compara a eficácia entre estes métodos como ponto de partida para o resto da dissertação, visando sempre a atualização dos processos de monitorização dos projetos através de metodologias ágeis, ou pelo menos híbridas.

Adicionalmente, destacou-se a secção 2.2 para um olhar atento sobre a qualidade dos processos e de produtos de software nas empresas, salientando-se os modelos CMMI e ISO9000 como guias para essas mesma qualidade.

Neste capítulo depreendem-se ainda, os conceitos teóricos de um indicador-chave de desempenho, que representa um valor mensurável que reflete uma estratégia ou objetivo de uma determinada empresa, bem como o quão longe a mesma estará de o cumprir [ver secção 2.3.2]. Uma métrica que também representa um valor mensurável mas que possui resultados diretos nos indicadores escolhidos também [ver secção 2.3.1]. E para finalizar, algumas considerações acerca de valores de *benchmarking*, o seu uso nas empresas, bem como o seu processo de implementação, caso surja a necessidade.

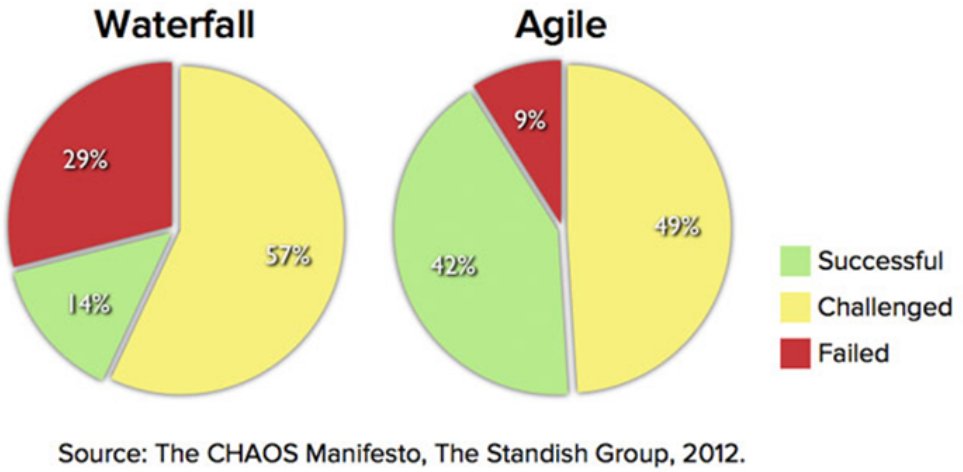

Figura 2.11: Comparação da eficácia dos métodos Agile e Waterfall. [7]

Em suma, foram apresentados os conceitos de KPI, dos métodos ágeis e em cascata, restando estabelecer relações entre eles e o trabalho a desenvolver. Como tal, é necessário analisar e definir indicadores que potenciem uma esperada transição suave e eficaz de uma cultura empresarial enraizada no método tradicional em cascata, para métodos mais ágeis no desenvolvimento de *software*, questões que serão abordadas nos próximos capítulos.

3. Melhoria dos Processos na KPI Consulting

O presente capítulo apresenta uma visão geral sobre a realidade inicial da empresa, os seus procedimentos, e as suas principais vicissitudes.

3.1 Questões em aberto

Após o esclarecimento acerca dos termos técnicos que alicerçam esta dissertação, o foco converge para as necessidades da KPI Consulting, a nível dos seus processos, e fundamentalmente, das suas equipas de desenvolvimento de *software*. Para o "reconhecimento do terreno", foram sendo realizadas deslocações à empresa do setor informático que proporcionaram não só um contacto pessoal com os membros das equipas que integram os projetos analisados, como um olhar atento sobre a realidade e forma de atuar das mesmas.

Primeiramente, deve-se salientar que tanto o processo de desenvolvimento, como o de controlo de qualidade se revelavam bastante *ah-hoc* (procedimento que não se encontra bem definido ou estipulado), na medida em que não se efetuavam medições que gerissem o estado do projeto, não se estruturavam bancos de dados históricos que permitissem comparações futuras, bem como não existia uma perceção do progresso qualitativamente (e quantitativamente também), entre outras problemáticas elucidadas de seguida.

Na fase inicial desta dissertação, os projetos S e P descritos na secção 1.1.1 e os seus respetivos colaboradores, articulavam uma cultura de produção de software contígua ao método em cascata. Esta metodologia, como abordado no capítulo anterior e através da figura 2.9, revela-se no geral, menos conseguida quando equiparada a desenvolvimentos híbridos ou ágeis. Em grandes organizações como a KPI Consulting, manifestou-se na necessidade de um responsável externo, analisar o processo de controlo de qualidade até então praticado. Deste modo, vendo a título de exemplo o projeto I, os seus intervenientes demonstraram interesse em assumir o compromisso de implementar práticas mais ágeis ao seu projeto, o que mais tarde viria a permitir a introdução de métricas e indicadores-chave de desempenho para o controlo contínuo do mesmo.

Por conseguinte, associada a uma mentalidade de produção "clássica" de *software*, a maior dificuldade enfrentada pela KPI Consulting reside no tempo estimado para a produção dos projetos que é consensualmente mais extenso do que na realidade se deveria mostrar. Esta afirmação deve-se ao facto do *Waterfall* não ser estruturado para as mudanças de requisitos durante o desenvolvimento do projeto, ou seja, à priori, as equipas sobrestimam o tempo da entrega para salvaguardar atrasos motivados pelas alterações de prioridades a meio.

De outro modo, como idiossincrasia do método em cascata, os projetos contactam unicamente com o cliente num período muito embrionário para a estipulação

dos requisitos funcionais e não-funcionais. Todavia, durante a fase de conceção a dificuldade em estabelecer meios fiáveis de comunicação (equipa-cliente), provocava uma indefinição, incerteza e frustração nos programadores durante a implementação de certas funcionalidades. Adicionalmente, surgem casos de precariedade na especificação dos requisitos que, aliada à débil comunicação com o cliente, contribuem para o tempo exacerbado que se estima para a conclusão dos projetos.

Para além disso, no caso do projeto S, a equipa de desenvolvimento está dependente de código legado por um dos programadores com mais experiência da empresa, o que na maioria dos casos se reflete na estagnação da implementação por certos períodos indefinidos, e que, num caso extremo (no caso de despedimento, ou ausência) pode levar o projeto ao seu término antecipado e inacabado.

Em suma, o conjunto destas vicissitudes levou, consequentemente, a KPI Consulting a atribuição desta dissertação a um responsável fora da mentalidade dos processos padronizados da empresa, para a implementação de medidas que controlem a qualidade e desempenho das suas equipas de desenvolvimento nos seus projetos.

3.2 Abordagem à Melhoria dos Processos

A principal demanda que estrutura a presente dissertação é: **"Sem um processo amadurecido, como medir, analisar e melhorar o desempenho das equipas de desenvolvimento de software, nomeadamente na KPI Consulting."**

A resposta prende-se com a introdução de métricas e indicadores-chave de desempenho, no processo de desenvolvimento dos projetos, porém não de uma forma abrupta nem invasiva para os colaboradores. Paralelamente, poderá dar-se a passagem para metodologias mais ágeis (ou pelo menos híbridas), as mesmas que viabilizam o uso de métricas e o consequente implante de indicadores-chave de desempenho no projeto em si. Após a introdução das métricas e KPI, o projeto é passível de ser monitorizado e guiado para efetuar as mudanças necessárias até atingir os resultados expectáveis.

De forma descritiva, a resolução para a interrogação destacada elucida-se nos seguintes passos [ver figura 3.1 para acompanhamento visual]:

1. Análise às vicissitudes do processo inerente às equipas de software que integram os diversos projetos na organização, por parte do responsável externo;

2. Identificação de possíveis métricas e indicadores-chave de desempenho que viabilizem um melhor desempenho das equipas, por parte do responsável externo;

3. Verificação por parte da KPI Consulting se de facto, as métricas e indicadores-chave de desempenho identificadas são as mais adequadas às necessidades e objetivos

4. Implementação, de acordo com as aprovações das equipas de desenvolvimento por cada projeto, das métricas e indicadores-chave de desempenho;

5. Produção de medições contínuas de dados relativos às métricas e indicadores-chave de desempenho validados para o projeto, e paralelamente, construir

um banco de dados com um histórico de valores que permitam assunções futuras;

6. Avaliação de todos os dados obtidos de modo a prosperar o objetivo proposto;

7. Consumação de um processo monitorizado e com o intuito de otimizar o desempenho das equipas de software adjacentes;

Por outro lado, deve-se ainda salientar que, não existiam quaisquer indicadores implementados, o que torna a sua escolha muito vasta e arriscada, uma vez que se pode estar a medir algo que não tenha efeito conclusivo no desempenho das equipas. Como tal, após a seleção dos KPI adequados, existe um processo de adaptação dos métodos de desenvolvimento ditos clássicos para os mais ágeis.

Espera-se então, que uma das maiores adversidades esteja assente na mudança de ambiente de desenvolvimento após migração, e a forma como irão lidar os membros das equipas com o seu novo mecanismo de desenvolvimento. Um dos passos será tornar regulares as reuniões abertas entre os membros das equipas, acerca da introdução de indicadores no seu desempenho, numa atmosfera de objetivos comuns e de correlação de ideias, e não de controlo.

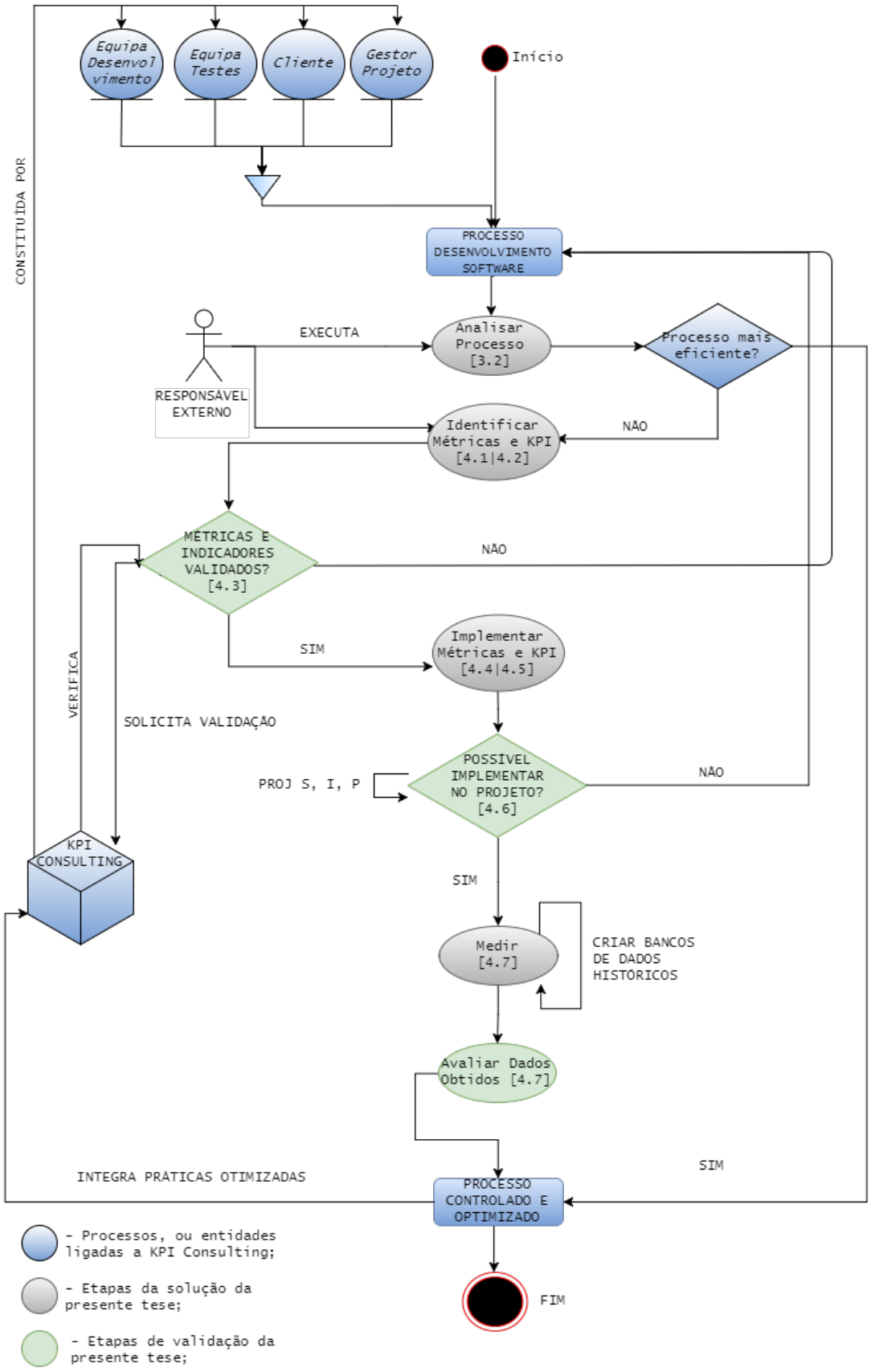

Figura 3.1: Modelo de Implementação da abordagem ao problema, com as diferentes fases elucidadas.

4. Indicadores-Chave de Desempenho nas Equipas da KPI Consulting

Neste capítulo evidenciam-se as métricas e os indicadores-chave de desempenho escolhidos como proposta de solução do problema.

4.1 Seleção de Métricas de Software

O processo de seleção de métricas de *software* foi tanto baseado em casos de empresas de IT que prosperam na construção de sistemas de gestão de desempenho[13], bem como nas necessidades maiores da KPI Consulting nesse ramo.

Por conseguinte, as métricas seguintes foram escolhidas não só por coadunarem com indicadores-chave de desempenho explicitados, mas também pela sua simplicidade de identificação, em detrimento de outras mais complexas. Isto é, são métricas de obtenção acessível e que facilmente se traduzem em variáveis úteis ao cálculo dos KPI .

De salientar ainda, que as métricas de *software* a utilizar selecionadas revelam-se um passo continuamente paralelo ao processo de escolha dos indicadores, uma vez que o uso de métricas por si só, é passível de manipulação por parte dos membros das equipas de desenvolvimento. Daí ambos se complementarem, as métricas no sentido quantitativo e de recolha de valor, e os indicadores na questão de guiar o processo para o caminho da melhor gestão. De seguida, são apresentadas sucintamente, as métricas eleitas.

4.1.1 Story Points

A métrica *"Story Points"*, visa expressar o esforço exigido às equipas de desenvolvimento para implementar determinada tarefa, ou corrigir um defeito encontrado.

Assim sendo, assim que conhecidos os requisitos, surgem as tarefas de cujo cumprimento e entrega revelam uma complexidade mais alta, e, através de uma reunião sobre o *sprint*, atribui-se a cada uma, uma categoria de dificuldade. Esta atribuição é feita, não só pela quantidade de trabalho a ser produzido, a sua complexidade de implementação, mas também pelo risco e incerteza do seu cumprimento.

Muitas das vezes, a complexidade de um defeito é tão elevada, que o mesmo se transforma numa nova tarefa, e aí, terá de se calcular o esforço associado à sua correção. Como tal, a métrica *Story Points*, também possui o intuito de padronizar e estabelecer medidas gerais e equitativas para todos os membros do projeto.

Esta métrica permite também associar a cada colaborador, a quantidade de trabalho executado, cumprido e atribuído, o que constitui uma mais valia na atribuição de regalias posteriores.

Frequência da medição: Por entrega (release);
Valor de referência: Depende da complexidade do projeto;

4.1.2 Número de Linhas de Código

A métrica *"Número de Linhas de Código"*, medida em LoCs ou KLoCs, visa contar o tamanho do sistema desenvolvido através das linhas que foram escritas para programar todas as suas funcionalidades. Assim que o banco de dados históricos seja concebido, esta métrica permitirá comparar esforços entre projetos semelhantes, e estimar, através do seu tamanho, conceções futuras.

De evidenciar que, no caso de gestão irresponsável de um projeto e à falta de indicadores que guiem esta métrica podem surgir ambiguidades na determinação do número de linhas de código[8]. Veja-se a título de exemplo os dois casos que se seguem:

```c
for (i = 0; i < 100; i++) printf("hello"); /* How many lines of code is this? */
```

Figura 4.1: Exemplo de linha de código na linguagem C.[8]

- 1 linha física;
- 2 linhas lógicas; (*for* e *printf*
- 1 linha de comentário

```c
/* Now how many lines of code is this? */
for (i = 0; i < 100; i++)
{
    printf("hello");
}
```

Figura 4.2: Excerto de código na linguagem C.[8]

- 4 linhas físicas;
- 2 linhas lógicas; (*for* e *printf*
- 1 linha de comentário

Assim sendo, para além da seleção desta métrica, também foi determinado um conjunto de predefinições através do seu ambiente de desenvolvimento integrado, que no caso das equipas de produção de software da KPI Consulting é o *Eclipse*. Estas delimitações são impostas pelo *Formatter e Cleanup* no CodeStyles da linguagem JAVA, e tornam o código de cada elemento da equipa idêntico ao da figura **4.2** para padronização do desenvolvimento.

Frequência da medição: Por entrega (release);
Valor de referência: Depende da complexidade do projeto;

4.1.3 Pontos de Função

A métrica *"Pontos de Função"* visa a contagem de funções existentes ao longo de todo o código que estrutura um sistema de *software*. Pode ser usada para mensurar o tamanho final do projeto, bem como para padronizar os esforços entre os colaboradores.

É uma técnica extremamente usual para empresas que, na sua génese, necessitam de valores mensuráveis para comparar, avaliar, planear e gerir a produção e tamanho do seu *software*.

Frequência da medição: Por entrega (release);
Valor de referência: Depende da complexidade do projeto;

4.1.4 Cobertura de Testes Unitários

A métrica *"Cobertura de Testes Unitários"*, visa medir a percentagem do *software* que está coberto por testes unitários, de modo a garantir a eficiência da equipa de testes (*testers*) e do código.

Primeiramente, é necessário que a equipa de testes que integra o projeto esteja a produzir testes unitários à medida em que é dado o desenvolvimento do produto e são entregues as tarefas (*US*) propostas. Estes testes unitários visam certificar que as funcionalidades e serviços estão implementados segundo o especificado nos requisitos.

Quanto maior for a cobertura dos testes unitários no código em questão, menor será a possibilidade de encontro falhas numa fase posterior. Sendo 100% o valor máximo que se pode atingir nesta métrica, estima-se que, as grandes empresas devam integrar valores acima dos 80%.[46] Porém, apesar de contraditório, a meta de atingir os 100% pode trazer problemas de manipulação de dados, pelo que esta métrica não se deve implementar sem o uso posterior de indicadores, na medida em que código que não esteja a passar nos testes, possa ser facilmente eliminado e/ou ludibriado.

Como se reflete num processo extremamente dispendioso, deve priorizar-se a cobertura de certos excertos de código que correspondem as funcionalidades mais meritórias

Frequência da medição: Por entrega (release);
Valor de referência: acima dos 80% [46] ;

4.1.5 Severidade das Falhas

Foi atribuída a métrica *Severidade das Falhas* às equipas de testes da KPI Consulting, visando atribuir um grau de complexidade aos defeitos encontrados aquando

da fase de teste do produto. Quanto maior for o impacto do erro encontrado no sistema, maior será a severidade que lhe é atribuída. Para além disso, quanto mais alta a prioridade, mais rapidamente se deverá a equipa concentrar na sua resolução.

Por conseguinte, as necessidades da KPI Consulting levaram a criação de quatro tipos de severidade descritos seguidamente:

Critical: O defeito leva a uma estagnação completa do processo, é aberta uma nova tarefa para que as equipas se concentrem na correção do mesmo,

High: Provoca o colapso de algumas funcionalidades do sistema e deve ser corrigido o quanto antes, é também aberta uma nova tarefa para que as equipas se concentrem na correção do mesmo,

Medium: O sistema permanece funcional, mas pode causar algumas alterações e comportamentos indesejáveis na produção.

Low: Não causa a interrupção do desenvolvimento, e a sua correção não deverá prolongar em demasia o projeto.

Type	ID	Linked Items	Story Points	Status	Severity
Defect	3847	0		Done	2-High
Defect	4721	2	5	Done	3-Medium
Defect	4749	0	5	Done	3-Medium
Defect	4567	1	5	Done	4-Low
Defect	4654	0	5	Done	3-Medium
Defect	4864	0	5	Done	4-Low
Defect	4911	0	5	Done	3-Medium

Table 4.1: Tabela de dados obtidos do projeto I, destacando a severidade das suas falhas.

4.1.6 Falhas no Código

A métrica *Falhas no código*, visa medir quantitativamente, o número de falhas (defeitos, bugs, erros) encontrados ao longo do desenvolvimento de cada plataforma.

À medida que se vão encontrando os erros, surge também a evidencia de que determinada área da aplicação a ser produzida precisa de mais atenção por parte dos seus criadores.

Frequência da medição: Mensal ou por entrega (release);

Valor de referência: Depende da complexidade e tamanho do *software* produzido;

4.1.7 Falhas corrigidas

A métrica *Falhas corrigidas*, visa medir quantitativamente, o número de falhas (defeitos, bugs, erros) corrigidas ao longo do desenvolvimento de cada plataforma, de acordo temporal sobre quando foram encontradas.

Esta medida, aliada à anterior, resolve a questão se a equipa de desenvolvimento da KPI Consulting encontra erros mais rapidamente do que os consegue corrigir, por exemplo.

Frequência da medição: Mensal ou por entrega (release);
Valor de referência: Depende da complexidade, das falhas no código encontradas e tamanho do *software* produzido;

4.1.8 Tempo efetivo para cada tarefa

A introdução da métrica *Tempo efetivo para cada tarefa*, tem o objetivo de medir temporalmente o esforço que cada interveniente do projeto precisa para a conclusão na íntegra da mesma. Quando se fala em conclusão na íntegra, estipula-se que só é dada como tal, quando a tarefa em questão apresenta todas as funcionalidades exigidas aquando da conceção dos requisitos do projeto.

Por conseguinte, no caso da equipa do projeto I, a KPI Consulting determinou que o tempo efetivo para cada tarefa será medido em horas, como pode ser observado na tabela que se segue:

Sprin	Type	ID	Actual (Hours	Done on dat
Sprint 1	User Story	4454	9	24-Nov-17
Sprint 1	User Story	4462	29	4-Dec-17
Sprint 1	User Story	3746	21	30-Nov-17
Sprint 1	User Story	2306	10	23-Nov-17
Sprint 1	User Story	2307	11	24-Nov-17
Sprint 1	User Story	4083	10	17-Nov-17
Sprint 1	User Story	4080	13	4-Dec-17
Sprint 1	User Story	4432	8	20-Nov-17
Sprint 1	User Story	3106	8	30-Nov-17
Sprint 1	User Story	3970	8	30-Nov-17

Table 4.2: Tabela referente ao projeto I, com o tempo efetivo de cada tarefa (US).

4.2 Seleção de Indicadores-Chave de Desempenho

Numa primeira abordagem, retiraram-se os Indicadores-Chave de Desempenho que se seguem através de obras bibliográficas reconhecidas como estruturais na enumeração de KPI, das quais se destaca: o COBIT[12] (Control Objectives for Information and Related Technologies), o guia de gestão da HP[13], a dissertação de Raj Ojha[47], a de Antolic[48] o manual publicado pelo Ministério de Planeamento, Orçamento e Gestão[49] e os guias da TaskManagementSoft[50].

Para a elucidação dos indicadores que se seguem, estrutura-se um modelo que integra um excerto descritivo de cada um, a sua fórmula de cálculo seguida do esclarecimento de possíveis variáveis que a representem, o valor de referência (benchmarking), e por fim, a frequência com que se deve medir.

Adicionalmente, os indicadores segmentam-se ainda em três divisões categóricas: *Indicadores Financeiros, Temporais e de Qualidade*, pelo que, se utilizaram as

siglas - "FIN", "TEM" e "QUA", antecedendo as dos KPI, para a ordenação segundo a sua categoria.

4.2.1 Custo do Esforço de um Empregado por Tarefa (FIN.CEET)

Este KPI representa uma das principais métricas que consentem o sucesso na melhoria do desempenho no desenvolvimento de software, na medida em que, permite obter uma noção do custo do trabalho de cada elemento das equipas que o constituem, por cada tarefa realizada.

No caso de necessidade da organização controlar, se de facto, o que pagam a um funcionário é justificado pela quantidade de tarefas que ele realiza, o CEET permite a medição de tais valores.

O cálculo deste indicador é efetuado através da equação:

$$CEET = (EA * VH) + (EADIC * VADIC)$$

Sendo que:

EA - Esforço concebido pelo empregado até à data para concluir tal tarefa, medido em story points [ver secção];

VH - Valor por hora de salário de cada empregado, pago pela empresa, medido em euros;

EADIC - Esforço extra ou adicional concebido pelo empregado para concluir tal tarefa, medido em storypoints;

VADIC - Valor por hora adicional de salário de cada empregado, pago pela empresa, medido em euros;

Valor de referência Depende do contexto da organização, base salarial e método de medição de esforço;

Frequência da medição: Por tarefa;

O valor obtido de cada medição do CEET representa diretamente os custos (em euros) para a organização por cada tarefa realizada por um empregado. Este KPI está correlacionado com o seguinte na questão do esforço adicional para se completar uma tarefa. De frisar ainda, que projetos com funcionalidades idênticas terão certamente tarefas e tempos de concretização muito fiáveis, daí o banco de histórico de dados para este indicador poderá trazer vantagens na padronização de projetos vindouros.

4.2.2 Custo de Reprogramar (FIN.CR)

O indicador *Custo de Reprogramar* verifica o número de horas suplementares, que uma equipa de desenvolvimento necessitou para a conclusão de uma ou mais funcionalidades. O CR alerta para o desempenho da equipa relacionado com os defeitos associados às tarefas produzidas, à eficiência da produtividade da mesma, e o seu tempo de resposta face às vicissitudes encontradas. Para além disso, foca-se no olhar atento sobre os custos que cada hora adicional provoca na organização.

O cálculo deste indicador é efetuado através da equação:

$$CR = \sum HR$$

Sendo que:
HR - Horas adicionais que todos os colaboradores tiveram de prestar para refazer determinada tarefa, sendo para implementação de novas funcionalidades exigidas ou para corrigir erros de severidade alta;
Valor de referência Depende do contexto da organização;
Frequência da medição: Por entrega (release);

De salientar ainda que, o valor obtido de cada medição do CR é calculado em horas, porém cada hora extraordinária traduz-se diretamente num custo (em euros) para a organização.

4.2.3 Desvio nos Custos do Projeto (FIN.DCP)

O indicador *Desvio nos Custos do Projeto* visa traduzir quantitativamente a questão de se o projeto, aquando da medição, está ou não dentro dos custos estimados. o DCP compara o custo que se estimou, na sua génese (fase de requisitos), que o projeto teria com o que durante o seu desenvolvimento ele tem vindo a custar.

Existem, por outro lado, alguns KPI equiparáveis ao DCP, como o *Lucro Obtido por Projeto*, entre outros, porém este oferece uma medição incremental durante o desenvolvimento da plataforma. Este facto, torna todos os dados da medição mais fidedignos e confiáveis através do olhar atento sobre os custos durante a fase de implementação.

Posto isto, o cálculo do DCP é efetuado através da equação:

$$DCP = \frac{CustoAtualProj - EstInicialCustoProj}{EstInicialCustoProj}.100\%$$

Sendo que:
TTE - Custo total e atual do projeto;
TTA - Estimativa inicial do custo do projeto;
Valores de Referência: DCP >1;
Frequência da medição: Semanal/Mensal;

Se DCP <1 significa que o projeto está a custar mais do que previamente foi estimado, por outro lado, caso DCP >1, então o mesmo apresenta características de progressão e de lucro no final.

Veja-se a título de exemplo prático:
Custo do projeto - 100,000€em 10 meses
Após a revisão mensal: O projeto está 15% feito com o custo de 18.000€
Ora, 15% x 100.000€= 15,000€
DCP = 15.000€/18.000€= 0,83
Então o projeto não está a ter sucesso a nível monetário.

Como elucidado no exemplo anterior, apesar do valor obtido de cada medição do DCP ser em percentagem, este reflete diretamente os custos (em euros) para a organização, e no caso de projetos semelhantes, pode-se recorrer aos valores anteriores deste KPI para prevenir determinados riscos e custos de projetos seguintes.

4.2.4 Custo de Mudança de Requisitos (FIN.CMR)

Numa primeira instância, as mudanças dos requisitos num método tradicional como o em cascata, aquando da fase de implementação, acarretam custos monetários avultados, representando um risco altíssimo para os lucros que o constituem. Por conseguinte, a implementação do CMR, bem como a sua medição tem como principal foco os projetos estruturados pelas metodologias ágeis uma vez que as alterações nos requisitos podem ocorrer, com alguma frequência.

O indicador *Custo de Mudança de Requisitos* mede-se, primeiramente, através do número de mudanças nos requisitos por parte do cliente, que foram aceites pela equipa de desenvolvimento, e numa segunda fase, pelo somatório dos custos associados a essas alterações.

Tal elucidado, o cálculo deste indicador é efetuado através da equação:

$$CR = \frac{CMRA}{CTP}.100\%$$

CMRA: Custo de Mudança nos Requisitos Aceites;
CTP: Custo Total do Projeto;
Valor de referência: Depende do contexto da organização;
Frequência da medição: Por Projeto;

De salientar que o CMR pode refletir alterações quantitativas no indicador anteriormente revisto na secção 4.2.3 - DCP. Nomeadamente no seu divisor, uma vez que ao valor CustoAtualProj terá que ser acrescido com o custo nas mudanças de requisitos. Em síntese, alguns dos KPI selecionados coadunam intrinsecamente com os outros de forma a que, o seu conjunto provoque uma escalada no desempenho das equipas de desenvolvimento de software. Para além disso, as mudanças nos requisitos aceites podem implicar alterações estruturais do sistema, levando à reformulação posterior na íntegra ou parcial do mesmo.

4.2.5 Entregar a Tempo (TEM.ET)

O indicador *Entregar a tempo* visa analisar se as funcionalidades planeadas por cada entrega ou release, são, de facto, implementadas atempadamente. Adicionalmente, pretende espelhar a capacidade de cada equipa que integra os diversos projetos de entregar os produtos funcionais nos tempos estimados.

Este KPI é proporcional ao apresentado em 4.2.9 - SC, na medida em que o incumprimento de prazos pode levar ao declínio da satisfação de clientes. As ferramentas dos métodos ágeis como a velocidade da equipa, e os "burndown charts", permitem estimar as entregas antecipadamente para que não existam derrapagens.

O cálculo deste indicador é efetuado através da equação:

$$ET = \frac{NFET}{NTF}.100\%$$

Sendo que:
NFET: - Número total de funcionalidades/release entregues a tempo, de acordo com o planeamento inicial;

NTF: - Número total de funcionalidades/release, de acordo com o planeamento inicial;

Frequência da medição: Semanal/Mensal;

De salientar o facto de que um ET alto significa que a empresa revela uma dificuldade enorme em cumprir prazos na entrega, o que pode por em risco tanto o serviço como a organização.

4.2.6 Cumprir Calendário (TEM.CC)

O indicador *"Cumprir Calendário"* mede a eficiência com que a equipa de desenvolvimento de software gere o seu tempo para as tarefas diárias mediante o tempo que se propôs para a sua realização.

Quanto menor for a percentagem melhor será o cumprimento do calendário, e consequentemente, está associada a melhoria do custo-eficiência e do tempo de execução do serviço. Uma CC de 100% é algo irrealista portanto as medidas de referência (benchmarking) para um serviço considerado eficiente, em grandes empresas, rondam valores entre 90% - 95% de adesão ao período calendarizado[51].

Estes valores refletem-se significativamente nos custos anuais da empresa, na satisfação do cliente (caso haja um atraso em demasia as entregas do produto), e no valor temporal estimado para cada tarefa das equipas de desenvolvimento. Veja-se a título de exemplo, um engenheiro informático que tenha uma falha na adesão ao calendário de 20 minutos diários, em cada tarefa executada, representa um total de 83 horas anuais, e um valor de aproximadamente 1000€, o que rapidamente se extrapola no caso de grandes empresas com mais de 100 empregados[51]. Para além disso, cada tarefa revela-se mais extensa temporalmente do que o devido e tal facto pode traduzir-se em atrasos nas entregas ao cliente, provocando uma menor satisfação.

No caso dos requisitos não se modificarem ao longo do período de desenvolvimento do projeto, o método de Waterfall apresenta, de facto um melhor **CC**. Porém, por outro lado, se os requisitos se alterarem ao longo do tempo, ou existirem modificações nas funcionalidades estruturais da plataforma, o KPI requer o uso de métodos ágeis para o seu melhor aproveitamento.

O cálculo deste indicador é efetuado através da equação:

$$AC = \frac{|TempoEfetivo - TempoEstimado|}{TempoEstimado}.100\%$$

Sendo que:
TempoEstimado: - Data planeada do térmido da entrega (release);
TempoEfetivo: - Data efetiva da entrega (release);
Valor de referência: [90%-95% [51];
Frequência da medição: Por entrega (release);

Em jeito de conclusão, o CC é um excelente propagador de bons hábitos de trabalho e está proporcionalmente interligado à eficiência na gestão do tempo das equipas de desenvolvimento de software.

4.2.7 Taxa de Tarefas Completas (QUA.TTC)

O indicador de eficiência *"Taxa de Tarefas Completas* visa evidenciar o número de tarefas que foram completas com todas as funcionalidades inerentes ao que estaria previsto nos requisitos.

Neste caso em específico, uma TTC baixa espelha a fraca capacidade de produção de software das equipas, bem como está subjacente ao não cumprimento do calendário, 4.2.6 CC, o que suscita atrasos na entrega. Isto é, quando mais alta a taxa de tarefas completas, melhor para a organização.

Num caso edênico, mede-se a TTC e o valor obtido é 100%, o que remete para um projeto com todas as funcionalidades implementadas, requisitos cumpridos, uma equipa extremamente eficiente, e um projeto pronto para lançamento com zero falhas. Por conseguinte, a TTC assenta em dois pilares estruturantes. Primeiramente, na **tarefa** completa com as funcionalidades inerentes, e em segundo, no quão "limpa" de erros está a mesma após a sua entrega.

Posto isto, a forma de cálculo deste indicador passa pela equação:

$$TTC = \frac{NTC}{NTT}.100\%$$

Sendo que:
NTC: - Número de tarefas completas;
NTT: - Número total de tarefas;

O caso de estudo de Jeff Sauro[9] sobre quase 1200 tarefas com utilizadores para as cumprir revela que, o valor de referência para este KPI serão os 78%, como se pode verificar através da imagem:

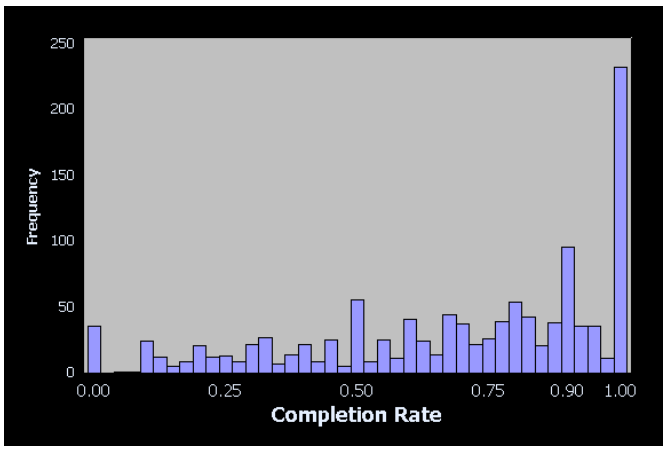

Figura 4.3: Distribuição de taxas de tarefas completas de 1200 tarefas. [9]

Valor de referência: Acima de 78%;
Frequência da medição: Por entrega (release);
Apesar dos valores de benchmarking serem conclusivos, estes dependem fundamentalmente do custo, complexidade, e prioridade da tarefa sendo que quanto maiores forem estes fatores, mais alta será a exigência adjacente ao TTC.

4.2.8 Taxa de Falhas (QUA.TF)

O indicador *Taxa de Falhas* visa indicar na totalidade das falhas encontradas (antes da entrega ao cliente), a percentagem das quais é que representam severidade **Critical** ou **High**, abordadas na secção 4.1.7. Defeitos no software deste grau, normalmente estagnam o processo até a sua resolução ser concretizada. Deste modo, no caso das medições da TF serem consideradas altas, isto pode representar que as tarefas não sejam entregues atempadamente, e que, consequentemente, haja um atraso em todo o processo. Este KPI está correlacionado com o SC - 4.2.9 , na medida em que o incumprimento de prazos pode levar ao declínio da satisfação de clientes.[10]

O processo de medição do TF também assegura que, de uma forma segura, são detetadas as falhas atempadamente.No que concerne ao período em que foi encontrada a falha, este afeta por conseguinte, a análise dos dados obtidos pelas medições do TF. Deste modo é perceptível que, um erro detetado na fase de conceção tem um custo de resolução baixo quando comparado com um erro detetado, por exemplo, pelo cliente aquando da entrega do sistema final.[10] Veja-se através da imagem que se segue, os custos da deteção de falhas, nas diferentes fases do projeto:[10]

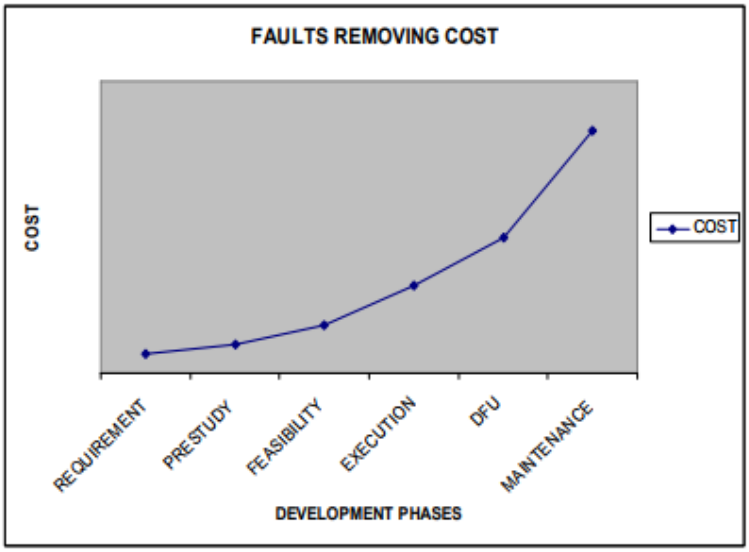

Figura 4.4: Custos da deteção de falhas, no TF, nas diferentes fases do projeto[10]

Tal exposto, a implementação da TF apresenta os seguintes benefícios: (i) - eficiência e deteção atempada de falhas críticas ou graves; (ii) - providencia dados sobre a severidade das falhas no sistema, alertando quem o analisa; (iii) - coordenação entre as equipas de desenvolvimento e de testes, que cooperam para um melhor desempenho; (iv) - obriga a prazos de entrega iterativos e mais curtos, visando a precisão aprimorada na entrega.[10]

O cálculo deste indicador é efetuado através da equação:

$$CD = \frac{NumFalhaCritHigh}{NumTotalFalha}.100\%$$

Sendo que:
NumFalhaCritHigh: - Número de falhas de severidade Critical ou High;
NumTotalFalha: - Número total de falhas;
Valor de referência: - <15%;[52]
Frequência da medição: Mensal;

Em síntese, a TF evidencia não só o rácio entre as falhas críticas ou graves do sistema e todas as falhas do sistema, mas também o período em que foi encontrada a falha. Deste modo, quanto mais tardiamente for encontrado, maiores serão os custos para o corrigir. De frisar ainda, que projetos com funcionalidades idênticas terão certamente tarefas e taxas de falhas muito fiáveis, daí que, um banco de histórico de dados para este indicador poderá trazer vantagens na padronização de projetos vindouros, e na estimação dos seus prazos.

4.2.9 Satisfação dos Clientes (QUA.SC)

O lucro empresarial é obtido através dos clientes, e como tal, saber se os clientes estão satisfeitos com o trabalho realizado é um passo crucial na prestação do serviço.

Gerir e cumprir com as expectativas do cliente, é o propósito primordial do indicador *Satisfação dos Clientes*. Ao desenvolver um produto de software, como referido em capítulos anteriores, este deve corresponder às características esperadas de qualidade do serviço, disponibilidade, facilidade de uso, eficiência, e o seu desempenho. Ao preencher estes requisitos, o valor da SC será, certamente, alto.[9]

O SC é medido recorrendo à American Customer Satisfaction Index (ACSI), que através de uma fórmula, todas as organizações podem comparar os valores obtidos por este KPI, com os principais rivais do seu ramo e indústria.[9]

	Base-line	06	07	08	09	10	11	12	13	14	15	16	17	Previous Year % Change	First Year % Change
All Others		75	75	75	76	77	79	77	76	77	74	81	79	-2.5	5.3
Computer Software		**74**	**73**	**74**	**75**	**76**	**78**	**77**	**76**	**76**	**74**	**81**	**78**	**-3.7**	**5.4**
Microsoft		73	70	69	70	76	78	75	74	75	75	80	76	-5.0	4.1

Table 4.3: Tabela de valores de referência do indicador Satisfação do cliente, por indústria.[11]

Como passível de verificar na figura, os valores nunca foram menores que 73% na indústria de *"Computer Software"*. Assim sendo, estipula-se esse valor como referência, o cálculo do indicador SC é efetuado através da equação:

$$SF = \frac{(Sat-1)*0.3885 + (Exp-1)*0.3190 + (Des-1)*0.2925)}{9}.100\%$$

Sendo que:

Sat: - Satisfação do cliente;
Exp: - Expectativa do cliente;
Des: - Desempenho do produto;
Valor de referênia: - >72%
Frequência da medição: Por entrega (release);

A título de curiosidade, por vezes este KPI, em projetos que usam metodologias ágeis, poderá também ser calculado através do medidor de felicidade (*Happiness Meter*), em que ao longo das *user stories* o cliente valida e classifica o seu estado de contentamento com o trabalho desenvolvido. Em síntese, o SC permite medir o nível de contentamento do cliente com a equipa que lhe desenvolveu o produto pedido, e após essa medição, visa comparar os resultados com os de outras empresas do mesmo sector.

4.2.10 Rácio de Defeitos (QUA.RD)

O indicador - *Rácio de Defeitos*, visa ressaltar o número de defeitos (erros, falhas, bugs, ou outro) encontrado, por cada 1,000 linhas de código escrito pelas equipas de desenvolvimento de software.

De acordo com Steve McConnell[53], os valores de referência da indústria da computação de software rondam intervalos de: [0-5] - bom desempenho; [5 - 15] - desempenho normal; [15 - 50] - desempenho fraco. Veja-se a título de exemplo, o caso da Microsoft, que após a medição do seu RD obteve o valor de 0.5 defeitos pela produção de cada 1,000 linhas de código. Ou da NASA que assume publicamente o seu RD é de 0. Todos estes valores apresentados sobre este KPI são o objetivo de todas as organizações mas, por outro lado, são **extremamente** dispendiosos. Para o alcance de medições como as da NASA, é necessário um investimento monetário por cada linha de código, e em muitos dos casos, cometer um erro dá direito a despedimento.[53]

A componente fulcral deste KPI é a medição constante e evolutiva do número de defeitos ao longo do crescimento do sistema. É possível, a partir desses números a visualização gráfica dos dados históricos sobre o RD, e prevenir eventuais falhas semelhantes num futuro próximo.

O cálculo deste indicador é efetuado através da equação:

$$RD = \frac{NumTotalDefeitos}{TamanhoSoftware}.100\%$$

Sendo que:
NumTotalDefeitos - Número total de defeitos encontrados;
TS - Tamanho de Software, em (KLoC);
Valores de referência: [0-5] defeitos/1,000 LoC;
Frequência da medição: Por entrega (release);

Em suma, obviamente o melhor resultado é 0%, porém é um objetivo demasiado dispendioso, e considerando todos os outros aspetos idênticos, um projeto maior tende a ter um maior RD do que um menor. O Tamanho Software pode ser medido nos termos da quantidade de linhas de código (LoC ou KLoC). E é de frisar ainda, que projetos com funcionalidades idênticas possuirão, de facto, tarefas e rácios de defeito muito fiáveis, daí que, um banco de histórico de dados para

este indicador poderá trazer vantagens na padronização de projetos vindouros, e na estimação dos seus prazos.

4.3 Verificação de Métricas e KPI

Após a seleção das métricas e indicadores-chave de desempenho mais adequados, através de uma reunião, foram apresentados à KPI Consulting um conjunto de 8 métricas e 11 indicadores, para que se validasse a sua utilidade, e complexidade de recolha desses mesmos dados na empresa. Assim sendo, seguem-se as tabela de resposta da KPI Consulting relativamente à possibilidade de medição dos mesmos nos projetos, e a verificação da existência de dados agregados históricos:

Indicador-Chave de Desempenho	Complexidade	Requisitos	Possível de medir?	Dados agregados
Custo do Esforço de um Empregado por Tarefa (CEET)	*ALTA*	Esforço Atual; Valor Hora; Esforço Adicional; Valor Adicional Hora;	**NÃO**	**NÃO**
Cumprir Calendário(CC)	*ALTA*	Tempo efetivo; Tempo estimado;	**SIM**	**NÃO**
Taxa de Tarefas Completas (TTC)	*MÉDIA*	Nº Tarefas completas; Nº Total tarefas;	**SIM**	**SIM**
Custo de Reprogramar(ER)	*ALTA*	Horas a Reprogramar;	**SIM**	**SIM**
Esforço de Desempenho (ED)	*ALTA*	Esforço estimado de trabalho feito; Esforço efetivo de trabalho;	**SIM**	**NÃO**
Taxa de Falhas (TF)	*ALTA*	Nº Falhas Testes Críticas ou Graves; Nº Falhas Total;	**SIM**	**SIM**
Satisfação de Clientes (SF)	*ALTA*	Grau de Satisfação; (1-10) Expectativa; (1-10) Desempenho; (1-10)	**SIM**	**SIM**
Rácio de Defeito (RD)	*ALTA*	Nº Total de Defeitos; Tamanho do Software (LoCs)	**SIM**	**SIM**
Desvio nos Custos do Projeto (DCP)	*ALTA*	Custo Atual Projeto; Estimativa Inicial Projeto;	**NÃO**	**NÃO**
Entregar a Tempo (ET)	*ALTA*	Nº Releases entregues a tempo (segundo o planeado); Nº Total de Releases	**NÃO**	**NÃO**
Custo de Mudança de Requisitos (CMR)	*ALTA*	(SUM) Custo Mudanças aceites;	**SIM**	**NÃO**

Table 4.4: Tabela de verificação de indicadores-chave de desempenho a usar por parte da KPI Consulting.

Métrica	Complexidade	Requisitos	Possível de medir?	Dados agregados
Pontos de Função	*BAIXA*	Número de Pontos de Função	**NÃO**	**NÃO**
Story Points	*BAIXA*	Esforço por cada tarefa	**SIM**	**SIM**
Linhas de código	*BAIXA*	Nº de linhas de código;	**SIM**	**SIM**
Testes unitários	*BAIXA*	Testes unitários escritos;	**SIM**	**SIM**
Cobertura dos testes unitários	*MÉDIA*	Testes são fidedignos;	**SIM**	**SIM**
Bugs no código	*BAIXA*	Nº de Bugs no código;	**SIM**	**SIM**
Bugs corrigidos	*BAIXA*	Nº de Bugs corrigidos;;	**SIM**	**SIM**
Tempo efetivo para tarefa	*MÉDIA*	Tempo real, não estimado.	**SIM**	**NÃO**

Table 4.5: Tabela de verificação de métricas a usar por parte da KPI Consulting.

4.4 Métricas de Software Adotadas

Nesta secção destacam-se as métricas que se pretendem implementar para um controlo qualitativo do processo e, por consequência, a maior eficiência das equipas de desenvolvimento. Essa prática foi precedida de uma verificação das métricas escolhidas de acordo com as necessidades prioritárias, do modelo de negócio, e dos objetivos da KPI Consulting.

As métricas escolhidas na secção 4.1, visam guiar um processo de desenvolvimento outrora baseado em métodos clássicos como o em cascata, para metodologias mais ágeis ou híbridas. Por conseguinte, de acordo com a tabela de verificação de métricas preenchida pela KPI Consulting - 4.5, destacam-se para implementação direta nos projetos não só as métricas, como também a justificação inerente a cada uma:

- **Pontos de História:** De forma a padronizar a medida de esforço extensível a todos os membros das equipas por cada tarefa, a KPI Consulting decidiu validar o conceito de *story points*, tornando assim mensurável a velocidade de desenvolvimento das mesmas. Os story points tentam obter uma aproximação razoável do esforço de cada colaborador com base no tipo de requisito. Assim sendo, os *story points* correspondem no fundo a três categorias: S(small = 5 story points), M (medium = 20 story points) e L (large = 50 story points). Esta estimativa é fornecida pela equipa na reunião de planeamento do *sprint*, com base na experiência adquirida pela equipa em *sprints* anteriores.

- **Número de Linhas de Código:** Com a validação do número de linhas de código, em *LoC's* ou *KLoC's* [ver Lista de Abreviaturas], a KPI Consulting, pretende observar de modo evolutivo, o tamanho de determinadas funcionalidades produzidas, bem como o tamanho total do software. Esta última, representa uma das variáveis a usar, por exemplo, no KPI destacado na secção 4.2.10.

- **Falhas no Código (encontradas e corrigidas):** Ao mensurar estas métricas, a empresa pretende controlar a nível do processo de desenvolvimento, as falhas e defeitos que são encontradas, por parte das equipas de testes, e a altura a que são encontradas, de modo a impedir atrasos significativos. De salientar que, estas métricas constituem variáveis construtivas nos KPIs das secções 4.2.7 e 4.2.8.

- **Severidade das Falhas:** De forma a categorizar as falhas encontradas pelas equipas de testes, em quatro graus diferentes, a KPI Consulting valida esta métrica para priorizar aquelas que estagnam o sistema (*Critical* ou *High*.

- **Tempo efetivo para cada tarefa:** Após a sua conclusão, o colaborador deve indicar qual o tempo efetivo que usou para executar determinada tarefa. Assim sendo, a validação desta métrica permite a KPI Consulting um maior controlo sobre os tempos de produção e entrega dos seus projetos.

As métricas descritas foram validados pela KPI Consulting através de requisitos fundamentalmente relacionados com: (i) - a necessidade que a empresa revela de um processo monitorizado e controlado; (ii) - uma gestão sustentável de execuções e prazos de entrega; (iii) - cálculo do número de falhas e o seu impacto; (iv) - necessidade de um processo iterativo que providencia o contacto entre a equipa e equipa-cliente; (v) - possibilidade de análise de dados mensuráveis com mais relevância para posterior atuação nas áreas mais problemáticas. De outro modo, restaram as métricas:

- **Pontos de Função -** uma vez constatado o uso de *Story Points* para medir o esforço das tarefas e de cada colaborador que as realiza, e o número de linhas de código escritas para declarar o tamanho total do projeto ao longo do seu desenvolvimento, a medição da métrica - ” Pontos de Função” não constitui uma prioridade para a KPI Consulting.

- **Cobertura de Testes Unitários -** apesar de cada *US* ser acompanhada de testes unitários, manuais e de regressão, por exemplo, com o objetivo de testar os serviços/funcionalidades implementadas, a métrica ”Cobertura de Testes Unitários” não constitui uma prioridade para a KPI Consulting na medida em que não é forçado que o projeto possua uma percentagem mínima de cobertura.

4.5 Indicadores-Chave de Desempenho Adotados

Nesta secção destacam-se os KPI que se pretendem implementar para um controlo qualitativo do processo e, por consequência, para a maior eficiência das equipas de desenvolvimento. De acordo com as necessidades prioritárias, do modelo de negócio, e dos objetivos da KPI Consulting surgiu a validação e ”filtragem” de alguns dos KPI identificados em detrimento de outros que não possuem características tão vigentes para a organização.

Os KPI evidenciados, não regulam só a produtividade, qualidade e usabilidade do processo de desenvolvimento. Os mesmos tornam passível de ser concretizado o trajeto de um procedimento outrora baseado em métodos tradicionais, para metodologias mais ágeis ou híbridas. Por conseguinte, de acordo com a tabela de verificação de indicadores-chave de desempenho preenchida pela KPI Consulting - 4.5, elucidam-se para implementação direta nos projetos não só os KPI, como também a justificação de utilização inerente a cada um:

- **Custo de Reprogramar -** Constitui um dos indicadores fundamentais para KPI Consulting, no sentido de controlar os gastos na produção "extraordinária" face à estimada inicialmente, a partir do número de horas adicionais usadas para cada tarefa ou na correção de falhas.

- **Custo de Mudança de Requisitos -** Como os KPI selecionados apontam na direção das metodologias ágeis, e como idiossincrasia das mesmas, existem algumas mudanças nos requisitos. Por conseguinte, este indicador permite o controlo dos custos adjacentes às mudanças aceites e à sua consequente implementação.

- **Cumprir Calendário -** Constitui um dos indicadores fundamentais, porém mais complexos de medir, para a KPI Consulting, uma vez que articula a entrega estimada, com a entrega efetiva. Ao longo do projeto, este indicador visa o cumprimento temporal das tarefas, colaboradores e *releases*, exercendo um controlo contínuo e ininterrupto do mesmo.

- **Taxa de Tarefas Completas -** A validação deste indicador representa para a KPI Consulting um conforto na verificação de se, de facto, as tarefas implementadas acarretam todas as funcionalidades especificadas pelos requisitos.

- **Taxa de Falhas -** A sua validação assenta no facto da KPI Consulting necessitar de valores mensuráveis entre o número total de falhas, e as que levam a estagnação do sistema, por parte das equipas de testes. Consoante os valores obtidos deste indicador, a empresa pode tomar medidas corretivas.

- **Rácio de Defeitos -** Constitui um dos indicadores fundamentais para a KPI Consulting, uma vez que a sua medição a posiciona comparativamente a valores na indústria do mesmo ramo. Ao longo do crescimento do tamanho projeto, este indicador visa destacar o número defeitos encontrados.

Os indicadores descritos foram validados pela KPI Consulting através de requisitos fundamentalmente relacionados com: (i) - a necessidade que a empresa revela de um processo monitorizado e controlado; (ii) - uma gestão sustentável de execuções e prazos de entrega; (iii) - cálculo do número de falhas e o seu impacto; (iv) - necessidade de um processo iterativo que providencia o contacto entre a

equipa e equipa-cliente; (v) - possibilidade de análise de dados mensuráveis com mais relevância para posterior atuação nas áreas mais problemáticas.

Por outro lado, rejeitaram-se os indicadores-chave de desempenho na medida em que:

- **Custo do Esforço de um Empregado por Tarefa** - representa uma complexidade elevada na mensurabilidade dos valores salariais por hora, uma vez que variam entre colaboradores dependendo do cargo, para além de que não se enquadra com as prioridades da KPI Consulting;
- **Desvio nos Custos do Projeto** - a sua mensurabilidade passa pela obtenção de dados extremamente sensíveis e confidenciais como valores (em euros) do custo total à empresa, fator de risco para a organização aquando da passagem para o responsável externo que a integra, daí a sua adoção não ter sido aprovada;
- **Entregar a Tempo** - em detrimento do uso deste indicador, optou-se pelo *Cumprir Calendário* uma vez que é um KPI mais geral e apresenta dados mais fiáveis no que concerne ao tempo de entrega estimado e efetivo;
- **Satisfação dos Clientes** - apesar de um KPI fundamental na avaliação da qualidade do processo, este não constitui peça estrutural nos interesses da KPI Consulting para esta dissertação, na medida em que a mesma tem o foco na gestão e melhoria de desempenho nas equipas de desenvolvimento de *software*.

4.6 Introdução dos Indicadores nos Projetos da KPI Consulting

Após a validação das métricas e indicadores-chave de desempenho [ver secção 4.4/4.5], por parte da KPI Consulting, segundos os seus requisitos e objetivos para a organização, deu-se a tentativa de introdução e implementação dos KPI nos projetos de *software* que decorriam na organização. De seguida, são destacados os projetos S, P e I para uma sucinta justificação sobre a concordância relativa à introdução dos KPI.

Projecto S

O projeto S que consiste na produção de uma *framework* de gestão e integração de identidades da KPI Consulting, é desenvolvido em *Waterfall*.

Apesar do projeto S apresentar algumas vicissitudes especialmente no que concerne ao código legado pelo decisor do projeto, e da escassa capacidade de reação à mudança nos requisitos, após a proposta de validação das métricas e indicadores-chave de desempenho identificadas, verificou-se que o mesmo, e os seus intervenientes não estariam interessados e de certa forma, aptos para uma mudança tão disruptiva na cultura dos seus processos e procedimentos.

Assim sendo, apesar de individualmente cada membro da equipa de desenvolvimento demonstrar interesse na implementação de metodologias ágeis, ou

pelo menos híbridas, o nível de compromisso com estas formas de trabalhar dependem sobretudo da estrutura organizacional da empresa e do serviço que gere o projeto (que pode não estar suficientemente motivado ou não ter os meios para levar a cabo o esforço dessa transformação).

Projeto P

O Projeto P que consiste na produção de uma aplicação Web usada por unidades de negócio para obter aconselhamento fiscal, é desenvolvido em *Waterfall*.

Por outro lado, inicialmente estaria estabelecida uma tentativa de adoção de métodos ágeis por vicissitudes associadas ao método em cascata, porém após a proposta de validação das métricas e indicadores-chave de desempenho identificadas, verificou-se que o projeto P, e os seus intervenientes não estariam interessados e de certa forma, aptos para uma mudança tão disruptiva na cultura dos seus processos e procedimentos.

Ou seja, apesar de individualmente cada membro da equipa de desenvolvimento demonstrar interesse na implementação de metodologias ágeis, ou pelo menos híbridas, o nível de compromisso com estas formas de trabalhar dependem sobretudo da estrutura organizacional da empresa e do serviço que gere o projeto (que pode não estar suficientemente motivado ou não ter os meios para levar a cabo o esforço dessa transformação).

Projeto I

O projeto I, que permite que os recursos humanos da KPI Consulting giram o processo de delegação a nível mundial dos seus colaboradores, foi o pioneiro na tentativa de adotar métodos ágeis, nomeadamente o *Scrum*, para a conclusão na íntegra do seu projeto.

Posto isto, a alteração da metodologia prende-se com o facto de ter passado a existir uma necessidade de evolução para métodos mais ágeis, e um compromisso em implementar a metodologia SCRUM. Este compromisso surgiu por parte da gestão do serviço da KPI Consulting.

Por conseguinte, este será o projeto protótipo na implementação das métricas e indicadores-chave de desempenho, na medida em que, as mesmas foram selecionadas e posteriormente validadas com o intuito de serem introduzidas em projetos com características ágeis, ou híbridas pelo menos. A opção tomada por métodos mais atuais, foi levada a cabo pela necessidade de comunicação contínua entre os intervenientes do projeto, bem como a monitorização do mesmo, que só é passível através da implementação dos indicadores. A equipa integrante do projeto I conta ainda com uma ferramenta de gestão para o *Scrum*, onde são introduzidas as tarefas, falhas e tempo estimados.

Em suma, as medições efetuadas, bem como as áreas de melhoria identificadas ao longo da presente dissertação [ver secção 4.7], são referentes ao projeto I e à sua equipa de desenvolvimento.

4.7 Medições Efetuadas e Valores Obtidos

As medições contínuas dos indicadores-chave de desempenho selecionados [ver secção 4.1/4.2] e posteriormente validados [ver secção 4.4/4.5] para introdução direta nos projetos representam um prosseguimento estruturante na presente dissertação.

Como referido na secção 4.6 deste capítulo, a obtenção de valores dos indicadores decorreu, dependendo das suas frequências, exclusivamente no projeto I, pelas suas características aproximadas às metodologias ágeis e ao controlo de processo. Assim sendo, os resultados obtidos, apesar de passíveis de assunções e estimativas que os correlacionem com projetos futuros idênticos, focam-se na melhoria do desempenho das equipas de desenvolvimento que integram o projeto I.

De acordo, com as especificações de cada indicador e as frequências de medição associadas, foram-se criando históricos de dados agregados, passíveis de consulta através do ANEXO A, obtidos pela ferramenta especificada na descrição do projeto I [ver secção 4.6 (Projeto I)]. As medições e consequentes dados agregados foram sendo recolhidos ao longo deste período de dissertação, de forma iterativa nas reuniões com a KPI Consulting, tendo surgido a sua génese no dia 1 de Setembro de 2017, e o culminar no 1 de Janeiro de 2018. Doravante todos os resultados obtidos e suas consequentes justificações provêm de dados sobre o projeto I, um dos pioneiros no processo de adoção de metodologias ágeis na KPI Consulting.

Para cada um dos indicadores que se seguem, estrutura-se um modelo de elucidação dos cálculos, de acordo com a sua frequência de medição, com as datas relativas a cada um, os valores das suas variáveis, o resultado obtido, uma análise dos dados com uma opinião sobre o indicador, a consequente avaliação em diferentes graus (A - Muito bom, B - Bom, C - Médio, D - Mau, e E - Muito mau.) A avaliação que se segue é estritamente relacionada com os valores de referência obtidos, sendo que a nota mais elevada representa valores acima dos de referência.

Posteriormente, esta secção providencia uma tabela com os indicadores, resultados obtidos para cada um deles de acordo com a sua frequência de medição, e um avaliador visual do estado em que se encontram cada um deles, para observação imediata do trabalho que foi realizado.

FIN.CR

Para o cálculo do indicador *Custo de Reprogramar* [ver secção 4.2.2] são necessárias duas variáveis: o número de tarefas que foram reabertas para correção ou para introduzir funcionalidade adicional, e o tempo, em horas que cada uma demorou a mais do que o estimado. Seguem-se os valores para ambas as *releases*, cada uma de dois meses de duração:

RELEASE 1.
[1Setembro-31Outubro]

Número de tarefas que excederam o seu tempo estimado: 97 tarefas

Segundo a equação elucidada na secção 4.2.2, o CR é calculado pelo somatório de horas que todas as tarefas excederam e por isso:

$$CR = 401horas$$

RELEASE 2.
[01Novembro-31Dezembro]

Número de tarefas que excederam o seu tempo estimado: 101 tarefas

Segundo a equação elucidada na secção 4.2.2, o CR é calculado pelo somatório de horas que todas as tarefas excederam e por isso:

$$CR = 441 horas$$

Análise dos dados obtidos: Na primeira entrega assiste-se a um total de 97 tarefas "extraordinárias" que correspondem a mais 401 horas de trabalho por parte das equipas de *software*. No que concerne à segunda entrega, verifica-se um total de 101 tarefas extra, que correspondem a mais 441 horas de trabalho por parte das equipas de *software*.

Assim sendo, o CR mostra que cada programador do projeto I, trabalhou mais 67 horas do que o inicialmente estimado, entre tarefas que foram reabertas, e falhas encontradas no sistema aquando dos testes de aceitação. Para a KPI Consulting, cada hora extra de um funcionário tem um valor associado (confidencial), mas a multiplicação do mesmo pelas horas evidenciadas, indica o custo avultado que a organização tem, por entrega, só com refazer código.

Devido à inutilização da mensurabilidade deste indicador e da ausência de um processo monitorizado e controlado, os custos associados à produção "extra" passavam ao lado do gestor do projeto. Adicionalmente, o mesmo fator evidencia as vicissitudes de um planeamento prévio desorganizado e uma falta de comunicação contínua entre os intervenientes do projeto.

Para além disso, as horas excessivas a "reprogramar" o sistema afetam proporcionalmente o custo final do projeto, e o cumprimento do calendário do mesmo.

Avaliação: D

Melhorias: Baseada na análise aos dados obtidos do indicador, surge a proposta de melhorias para o desempenho das equipas de desenvolvimento de software prosperar. Numa fase embrionária, deve ser bem definido um planeamento inicial da arquitetura do sistema, com as funcionalidades prioritárias destacadas para uma maior atenção às mesmas por parte de todos os membros integrantes do projeto. O gestor do projeto, através dos valores obtidos, deve desempenhar um papel ativo e atento não só como guia para os objetivos, mas também como alerta sobre a eficiência da sua equipa. A implementação de reuniões de revisão destes valores pode positivamente levar a um maior entrosamento entre membros do projeto (discussão cara-a-cara). Adicionalmente, devem ser estabelecidas linhas de comunicação eficazes para a redução dos intervalos de tempo em que se deteta uma falha, o tempo em que se começa a corrigi-la, e o tempo em que se acaba de a corrigir.

TEM.CC

Para o cálculo do indicador *Cumprir Calendário* [ver secção 4.2.6] são necessárias duas variáveis: o tempo efetivo de entrega da *release*, e o tempo estimado para essa mesma entrega. Apresenta-se visualmente na figura 4.5, o planeamento da KPI Consulting, para o projeto I a nível de entregas, seguido dos valores obtidos para as *releases*:

RELEASE 0.
[01Julho-01Setembro]

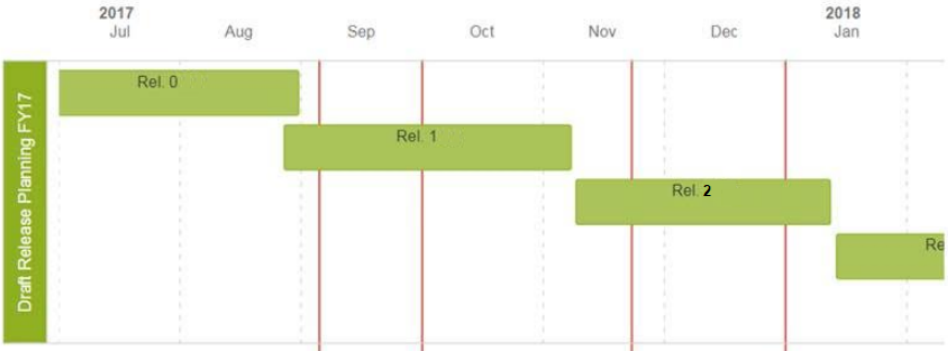

Figura 4.5: Planeamento de entregas do projeto I da KPI Consulting. [7]

Data início da *release*: 01/07/2017
Data de entrega estimado: 01/09/2017
Data de entrega efetivo: 16/09/2017
TempoEstimado: 01/09/2017 - 01/07/2017 = 62 dias.
TempoEfetivo: 16/09/2017 - 01/07/2017 = 77 dias.
Segundo a equação elucidada na secção 4.2.6:

$$CC = 80,51\%$$

RELEASE 1.
[01Setembro-01Novembro]

Data início da *release*: 01/09/2017
Data de entrega estimado: 01/11/2017
Data de entrega efetivo: 25/11/2017
TempoEstimado: 01/09/2017 - 01/11/2017 = 61 dias.
TempoEfetivo: 25/11/2017 - 01/09/2017 = 85 dias.
Segundo a equação elucidada na secção 4.2.6:

$$CC = 71,8\%$$

RELEASE 2.
[01Novembro-01Janeiro]

Data início da *release*: 01/11/2017
Data de entrega estimado: 01/01/2018
Data de entrega efetivo: 16/09/2017
TempoEstimado: 01.01.2018 - 01/09/2017 = 62 dias.
TempoEfetivo: 14.01.2018 - 01.09.2017 = 74 dias.
Segundo a equação elucidada na secção 4.2.6:

$$CC = 83,7\%$$

Análise dos dados obtidos: Na primeira medição assiste-se a uma discrepância entre o tempo estimado de entrega, e o efetivo de 15 dias, representando um CC de 80,51%. No que concerne à segunda medição, como efeito bola

de neve pela diferença na primeira medição, obteve-se uma diferença de 24 dias entre, correspondendo a um CC de 71,8% e na última medição, uma diferença de 12 dias, que correspondeu a um CC de 83,7%. Este último valor, representa um cumprimento do calendário muito bom no sentido em que a altura da segunda *release* existiu uma melhoria na velocidade de desenvolvimento, recuperando alguns dias, e, para além disso apanhou a fase final do ano, o que corresponde a muitos dias livres.

Por conseguinte, apesar de todas as medições apresentarem valores de cumprimento do calendário superiores a 70%, o que, para um indicador que nunca tinha sido medido na organização, é bastante razoável, para uma organização da reputação da KPI Consulting, estes valores devem ser motivo de atenção e de melhoria contínua. A monitorização deste indicador, acarreta uma ferramenta de controlo corretivo sobre como os colaboradores estão a usar o tempo na empresa, aos olhos do gestor de projeto.

Para além disso, cria também uma cultura assente na igualdade na medida em que todos os períodos contam, e os colaboradores devem ser responsáveis pelo quão produtivos são, em determinado tempo gasto, porém com a monitorização deste KPI, visa-se que a produtividade seja aumentada e o tempo gasto diminuído em proporção, mas equiparável entre todos os intervenientes no desenvolvimento.

Como tal, os atrasos verificados numa primeira instância podem não só ocorrer devido à falta de monitorização da altura, por parte do gestor de projeto, sobre os horários (chegada e saída) da equipa de desenvolvimento, mas também a falhas na comunicação entre os intervenientes do projeto.

Adicionalmente, os dias excedidos na entrega representam custos para a organização.

Avaliação: D

Melhorias: De forma a aumentar a produtividade e eficiência das equipas de desenvolvimento de *software*, para cumprimento maior de prazos estipulados, a KPI Consulting poderá conceder prémios de reconhecimento, não só para congratular os que cumpriram com as tarefas e prazos estimados, mas também para motivar os que não o fazem. Para além desta medida, com a agregação de valores históricos, o gestor de projeto pode prever, temporizar e realçar quem são os membros do projeto em destaque, e a sua disponibilidade ao longo do desenvolvimento.

QUA.TTC

Para o cálculo do indicador *Taxa de Tarefas Completas* [ver secção 4.2.7] são necessárias duas variáveis: o número de tarefas que foram completadas com todas as funcionalidades subjacentes, e o número total de tarefas. Seguem-se os valores para ambas as *releases*, cada uma de dois meses de duração:

RELEASE 1.
[01Setembro-31Outubro]

Número de tarefas completas: 81

Número total de tarefas: 96

Segundo a equação elucidada na secção 4.2.7, o TTC é calculado pela divisão do número de tarefas completas, pelo total de tarefas e por isso:

$$TTC = 84,38\%$$

RELEASE 2.
[01Novembro-01Janeiro]

Número de tarefas completas: 86
Número total de tarefas: 91
Segundo a equação elucidada na secção 4.2.7, o TTC é calculado pela divisão do número de tarefas completas, pelo total de tarefas e por isso:

$$TTC = 94,5\%$$

Análise dos dados obtidos: Na primeira entrega, o valor do TTC é de 85,38% justificado pelo facto de somente 15 tarefas não terem sido totalmente completas. No que concerne à segunda entrega, denota-se um crescimento significativo de tarefas completas, na medida em que o valor do TTC subiu para os 94,5%, e em 91 tarefas, apenas 5 não ficaram completas.

Sendo os 100% a meta esperada, os valores obtidos nas entregas do projeto I, para resultados pioneiros neste indicador no contexto empresarial, revelam-se bastante razoáveis. Atendendo aos valores de *benchmarking* elucidados pela figura 4.3, ambos os resultados se qualificam de forma superior aos 78% de média, e, como tal, quanto as tarefas que foram completas a KPI Consulting e as suas equipas de desenvolvimento revelam uma excelente competência no que concerne à entrega das tarefas completas com as funcionalidades propostas.

Uma TTC alta demonstra uma qualidade na produção fora do normal, e está proporcionalmente relacionada com a reputação da empresa.

Avaliação: A

Melhorias: Há sempre espaço para melhoria, e, apesar dos valores obtido serem altos, existem melhorias adjacentes à TTC que podem ser implementadas para a aproximação aos 100%. A partir deste indicador, deve-se instaurar a mentalidade de eficiência extensível a todos os membros da equipa, e não de controlo absoluto por parte do gestor de projeto, uma vez que, assim todos visam terminar as suas tarefas e estarem no mesmo patamar. Através de reuniões de revisão, dar voz aos intervenientes do projeto que não cumpriram com a conclusão integral de determinada tarefa, de modo a recolher *feedback* sobre as vicissitudes que os levaram a tal acontecimento inesperado.

QUA.TF

Para o cálculo do indicador *Taxa de Falhas* [ver secção 4.2.8] são necessárias duas variáveis: número de falhas de severidade crítica ou grave (Critical ou High), e o número total de falhas. Devido à sua frequência de medição, seguem-se os valores obtidos para todos os meses:

Setembro

Número de falhas *Critical* ou *High*: 25
Número total de falhas: 54

Segundo a equação elucidada na secção 4.2.8, a TF é calculada pela divisão do número de falhas *Critical* ou *High* pelo número total de falhas, e como tal:

$$TF = 46\%$$

Outubro

Número de falhas *Critical* ou *High*: 24

Número total de falhas: 43

Segundo a equação elucidada na secção 4.2.8, a TF é calculada pela divisão do número de falhas *Critical* ou *High* pelo número total de falhas, e como tal:

$$TF = 55,8\%$$

Novembro

Número de falhas *Critical* ou *High*: 17

Número total de falhas: 59

Segundo a equação elucidada na secção 4.2.8, a TF é calculada pela divisão do número de falhas *Critical* ou *High* pelo número total de falhas, e como tal:

$$TF = 28,8\%$$

Dezembro

Número de falhas *Critical* ou *High*: 17

Número total de falhas: 54

Segundo a equação elucidada na secção 4.2.8, a TF é calculada pela divisão do número de falhas *Critical* ou *High* pelo número total de falhas, e como tal:

$$TF = 31,5\%$$

Análise dos dados obtidos: Nas primeiras duas medições efetuadas, relativas aos meses de Setembro e Outubro, denota-se que a percentagem de falhas que levam a interrupção do sistema até sua resolução (Severidade: *Critical* ou *High*), é aproximadamente metade de todas as outras. Já a medição de Novembro apresenta resultados inferiores (28,8%) sendo que Dezembro, aumenta para os 31,5%, porém resultados inferiores ás duas iniciais.

Ainda que tenham sido alvo de melhorias, os resultados obtidos são considerados elevados atendendo aos valores de referência. Porém, observando o gráfico de custos temporal da deteção de erros, é de salientar que estes estão a ser detetados na fase de execução, o que suaviza o impacto negativo obtido pela medição do indicador TF. Quanto mais cedo forem detetadas as falhas, menor será o custo associada às mesmas.

Como tal, estes resultados ressalvam que a implementação de um sistema que monitorize o desempenho das equipas de desenvolvimento, acarretam melhorias inatas ao processo (como é de observar pelos valores ao longo das medições).

Avaliação: D

Melhorias: Os resultados obtidos alertam para a melhoria especificamente em áreas como: revisão de código antes de entrega, bem como o uso de um

processo mais formal para testes unitários, padronização do esforço ao longo do projeto através de um planeamento devidamente estruturado, na medida em que existe uma discrepância elevada entre determinadas fases do projeto.

FIN.CMR

Para o cálculo do indicador *Custo das Mudanças nos Requisitos* [ver secção 4.2.2] são necessárias duas variáveis: o custo das mudanças no requisitos aceites, e o custo final do projeto.

Apesar do CMR representar um indicador fulcral para a KPI Consulting na medida em que, afere a percentagem do custo final de um projeto verificada pelas mudanças aceites nos requisitos, no caso do projeto I as medições não foram efetuadas. A forma sobre a qual as equipas de desenvolvimento trabalham insere-se na medida em que, aquando do surgimento de novas modificações no sistema, estas diluem-se através das tarefas afetadas, descaracterizando tal facto, como um mudança nos requisitos.

Por conseguinte, a alteração da realidade relativamente à anotação de uma mudança nos requisitos, não foi conseguida, sendo o impacto dos custos associados posto de parte.

QUA.RD

Para o cálculo do indicador *Rácio de Defeito* [ver secção 4.2.10] são necessárias duas variáveis: o número de defeitos que foram encontrados ao longo do desenvolvimento por parte das equipas de testes, e o tamanho do *software* produzido, a cada *release*.

RELEASE 1.
[1Setembro-31Outubro]

Número de Defeitos que foram encontrados: 156
Tamanho Software: 134 *KLoC's*
Segundo a equação elucidada na secção 4.2.2, o RD é calculado pela divisão do número de defeitos que foram encontrados, pelo tamanho de software, e como tal:c

$$RD = 1,164 Defeitos/1 KLoc$$

RELEASE 2.
[01Novembro-01Janeiro]

Número de Defeitos que foram encontrados: 124
Tamanho Software: 1345*KLoC's*
Segundo a equação elucidada na secção 4.2.2, o RD é calculado pela divisão do número de defeitos que foram encontrados, pelo tamanho de software, e como tal:c

$$RD = 0,92 Defeitos/1 KLoc$$

Análise dos dados obtidos: O valor da primeira medição do RD é de 1,164 defeitos por cada mil linhas de código escrito sendo que o tamanho de *software*,

nesta altura, seria de 134 000 linhas de código. No que concerne à segunda medição, o valor é de 0,92 defeitos por cada mil linhas de código sendo que o tamanho de *software*, nesta altura, seria de 135 000 linhas de código. A questão do projeto só ter aumentado 1000 linhas de código entre as duas entregas prende-se com o facto de existirem "limpezas" no código regulares na KPI Consulting, de modo a evitar duplicação, bem como linhas extra desnecessárias.

Deste modo, ambos os valores apresentam-se bastante residuais atendendo à complexidade e tamanho do *software* produzido. Quando equiparado aos valores de referência explicitados pela secção 4.2.10, os valores obtidos compreendem-se dentro do intervalo. Demonstrando assim que, o número de falhas encontradas pela equipa de testes durante o desenvolvimento, são residuais dada a arquitetura do projeto em si.

Avaliação: A

Melhorias: Sendo o objetivo sempre a minimização dos erros encontrados, até atingir os 0, devem elucidar-se algumas melhorias nesta questão, apesar dos valores se refletirem numa organização que prospera. Devem ser apresentados, de forma transparente, o impacto dos custos que cada defeito evidencia, de modo a consciencializar todos os intervenientes numa produção de excelência, dado o prestígio da empresa para a qual trabalham. Para além disso, devem-se repartir equitativamente o esforço de determinadas tarefas, para uma entrega mais faseada e, consequentemente, a verificação de possíveis defeitos na produção do *software*, por parte da equipa de testes.

4.8 Medidores de Validação de Resultados

Os resultados obtidos ao longo das medições dos indicadores-chave de desempenho escolhidos para introdução nos projeto I da KPI Consulting, refletem áreas passíveis de serem melhoradas no que concerne às equipas de desenvolvimento.

Como explicitado através da tabela 4.6, os indicadores-chave de desempenho "Rácio de Defeitos", e "Taxa de Tarefas Completas", após a sua validação e através das suas medições com frequência por *release*, demonstram uma recetividade excelente por parte da equipa de desenvolvimento e resultados avaliados elevados. Por outro lado, KPI como "Cumprir o Calendário, "Custo de Reprogramar", "Taxa de Falhas", apresentam resultados não tão favoráveis e revelam a necessidade notória de melhorias, que foram apresentadas no capítulo 4.7. No que concerne ao "Custo de Mudança de Desempenho", apesar da sua validação por parte da KPI Consulting, não existiu uma adaptação da mentalidade das equipas de desenvolvimento de *software*, pelo que não chegou a ser medido.

Por conseguinte, a tabela 4.6, permite visualizar os medidores por forma a alertar quais serão os que precisam de maior atenção, bem como um panorama geral sobre o estado dos processos e dos serviços da empresa.

Indicador-Chave de Desempenho	Tipo de Medição	M1 - SEP	M2 - OCT	M3- NOV	M4 - DEC	Resultado
		Release 1		Release 2		
TEM.CC	*Release*	**71,8%**		**83,7%**		
QUA.TTC	*Release*	**84,2%**		**94,5%**		
FIN.CR	*Release*	**401 horas**		**441 horas**		
QUA.TF	*Mensal*	**46%**	**55,8%**	**28,8%**	**31,5%**	
QUA.RD	*Release*	**1,164** Defeitos por cada 1,000 linhas de código;		**0,92** Defeitos por cada 1,000 linhas de código;		
FIN.CMR	*Projeto*					

Table 4.6: Tabela de validação de indicadores e medidores de desempenho das equipas de desenvolvimento da KPI Consulting.

5. Conclusões

Neste capítulo, elucidam-se as implicações da presente dissertação a nível empresarial, bem como os objetivos que foram alcançados e as perspetivas de trabalho futuro.

5.1 Conclusão

O futuro das equipas de desenvolvimento da KPI Consulting pode potenciar-se, de acordo com o exposto no documento, pela monitorização do seu processo através de indicadores-chave de desempenho.

Por conseguinte, a implementação dos indicadores escolhidos viabiliza uma transição dos métodos "tradicionais" em que se encontram profundamente enraizados a maioria dos projetos da empresa, em métodos ágeis ou híbridos mais atuais, apesar de ambos revelarem vantagens na sua implementação.

No caso do projeto I, esta metamorfose implicou mudanças nos intervenientes dos projeto no que concerne à sua forma de pensar, foco de atuação, no planeamento, na comunicação entre membros de equipa e clientes, e membros entre si próprios. Na medida em que, foi o projeto pioneiro a integrar metodologias mais ágeis para a produção de *software* na KPI Consulting. Por outro lado, este método de desenvolvimento anuiu a integração dos indicadores-chave de desempenho validados e a sua consequente avaliação para colocar em destaque as áreas de melhoria mais proeminentes.

Em conjunto com os valores obtidos ao longo do período de medições, agregaram-se as áreas e estratégias que efetivamente necessitavam de melhorias. Entre as quais, a nível da génese e arquitetura dos projetos, deve ser bem definido um planeamento com as funcionalidades prioritárias destacadas; o gestor do projeto deve exercer uma posição ativa e disposta a cumprir com todos os detalhes para o bom funcionamento do processo; os meios de comunicação entre os todos os intervenientes do projeto devem ser claras e estáveis, de modo a que o contacto seja contínuo e inefável e, por exemplo, através da monitorização e do controlo dos processos, os próprios colaboradores se sentem destacados para uma eficiência maior na implementação visando alcançar os melhores resultados da organização. Ora, através das medições efetuadas, de acordo com os indicadores, ressalvaram-se propostas de melhoria associadas aos mesmos, perspetivando sempre o aperfeiçoamento do controlo do processo

Em suma, existem vários fatores que influenciam o desempenho de uma organização no processamento dos seus produtos tais como, a cultura empresarial enraizada, o quão aperfeiçoado, monitorizado e maturado está o método de desenvolvimento, a cooperação e comunicação entre todos os membros do projeto, e o ambiente em que se executam as tarefas.

5.2 Objetivos Alcançados

De um modo sucinto, os objetivos alcançados verificam-se através do cumprimento, ou não, de todas as especificações propostas pela secção dos resultados esperados [ver secção 1.5].

Primeiramente, foi efetuada a pesquisa, dentro dos que existem, e consequente definição dos indicadores-chave de desempenho relevantes de acordo com as tipologias dos projetos e os objetivos de suporte a melhoria continua. Em segundo plano, validou-se com a organização, a escolha anterior, de forma a integrar os indicadores que especificamente se revelam uma mais valia a novel de monitorização e controlo da qualidade de processo e de produto. De seguida, definiram-se os modos de recolha dos dados, em conformidade com as equipas de desenvolvimento da KPI Consulting. Como quarta etapa, evidenciou-se a identificação/obtenção de *baselines* e *benchmarks* que permitiram estabelecer valores de referência para os indicadores-chave de desempenho destacados em fases anteriores, possivelmente dependentes de características dos projetos. E por ultimo, salientaram-se pontos fulcrais de melhoria nas áreas mais críticas com base nas métricas e valores obtidas de forma contínua mensalmente ou pelas diferentes entregas dos projetos.

Por conseguinte, e através do exposto, foi alcançada: a lista de métricas de software que viabilizam a implementação dos indicadores-chave de desempenho; a lista de indicadores-chave de desempenho que visam a melhoria do processo da empresa em questão; os dados mensurados das diferentes áreas que caraterizam uma equipa de desenvolvimento, para objetivos de melhoria contínua das mesmas nos seus projetos.

No que concerne à migração dos projetos da KPI Consulting, mais concretamente nos projetos S e P, essa passagem para utilização de metodologias mais ágeis não foi passível de ser concebida. Dado à complexidade organizacional e o esforço requerido para tal, só o projeto I adotou as medidas necessárias para ser o pioneiro a implementar as metodologias ágeis, mais concretamente o *Scrum*. Como tal, e sendo as métricas e indicadores-chave de desempenho escolhidos para adoção de métodos ágeis, estas só foram introduzidas na equipa de desenvolvimento que integra o projeto I.

Por outro lado, os dados analisados apesar já apresentarem algumas indicações, como exposto ao longo da secção 4.7, só foram obtidos segundo duas entregas efetuadas, durante o tempo desta dissertação. Como tal, para aferir as melhorias causadas pelos indicadores introduzidos no projeto I, por exemplo, a médio-longo prazo, seria necessária um período para execução da dissertação, consequentemente maior.

5.3 Trabalho Futuro

O trabalho futuro relativo à presente dissertação na KPI Consulting pode resultar não só na recolha de métricas para além das que concernem as equipas de desenvolvimento, bem como na automatização dos indicadores-chave de desempenho identificados.

Por um lado, é notória a necessidade de escolha de indicadores que meçam o desempenho dos outros setores do ramo da KPI Consulting, na medida em que a organização deve evoluir de forma sustentável no que concerne ao controlo e medição de indicadores em todas as áreas estruturantes.

Para além disso, ressalva-se a necessidade de, através da ferramenta a que os projetos da empresa recorrem, inserir os KPI escolhidos para que, paralelamente ao aumento das suas variáveis o valor do mesmo seja atualizado, através das fórmulas de cálculo de cada um.

Por conseguinte, com a automatização dos indicadores abordados ao longo desta dissertação, a KPI Consulting conseguirá, através das medições, uma monitorização instantânea da qualidade do seu processo de desenvolvimento. Dependendo da frequência de medição de cada indicador, os resultados obtidos poderão refletir que determinadas mudanças devem ocorrer, que o foco do projeto está a ser desviado ou mesmo que o colaborador A ou B não está a ser o produtivo suficiente para os padrões da empresa.

Referências

[1] H. Engolm, "Processos de desenvolvimento de sistemas." Agosto 2013. [Online]. Available: https://helioengholmjr.files.wordpress.com/2013/08/cascata.png

[2] "Metodologias Ágeis de desenvolvimento de software." [Online]. Available: http://www.brq.com/metodologias-ageis/

[3] "10th annual state of agile report." [Online]. Available: https://explore.versionone.com/state-of-agile/versionone-10th-annual-state-of-agile-report-2

[4] K. Schwaber, *Agile Project Management with Scrum*. Microsoft Press, 2004.

[5] I. SOMMERVILLE, *Software Engineering: (8th Edition) (International Computer Science)*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2006.

[6] G. Analytics, "Kpi analysis," 2016. [Online]. Available: http://unilytics.com/kpi-analysis/

[7] "Metodologias Ágeis." [Online]. Available: https://danielettinger.com/category/metodologias-ageis/

[8] "Software requirements and project management." [Online]. Available: http://teaching.csse.uwa.edu.au/units/CITS3220/readings/metricsHandout.pdf

[9] J. Sauro, "What is a good task-completion rate." [Online]. Available: https://measuringu.com/task-completion/

[10] D. Duka, *Fault Slip Through Measurement in Software Development Process*. Research and Development Center Ericsson Nikola Tesla, 2012.

[11] "Acsi - computer software." [Online]. Available: http://www.theacsi.org/industries/telecommunications-and-information/computer-software

[12] ISACA, *COBIT 5: A Business Framework for the Governance and Management of Enterprise IT*. ISACA International, 2012.

[13] HP, *Construção de um sistema de desempenho para as TI*. Hewlett-Packard Development Company, November 2011.

[14] W. W. Royce, *Managing the Development of Large Software Systems*. USA: IEEE Computer Society, August 2006.

[15] Tutorialspoint, "Sdlc - waterfall model." [Online]. Available: https://www.tutorialspoint.com/sdlc/pdf/sdlc_waterfall_model.pdf

[16] A. ALLIANCE, "Manifesto for agile software development." [Online]. Available: http://www.agilemanifesto.org

[17] M. SOARES, *Comparação entre Metodologias Ágeis e Tradicionais para o Desenvolvimento de Software.* IUnipac - Universidade Presidente Antônio Carlos, Faculdade de Tecnologia e Ciências de Conselheiro Lafaiete.

[18] M. Tomás, *Métodos ágeis: características, pontos fortes e fracos e possibilidades de aplicação.* IET Working Papers Series, 2009.

[19] "Scrumalliance – transforming the world of work." [Online]. Available: http://www.scrumalliance.org/

[20] R. E. Jeffries, "Xprogramming - an agile software development resource." [Online]. Available: http://www.xprogramming.com/

[21] "Rapid application development." [Online]. Available: https://www.outsystems.com/blog/rapid-application-development.html

[22] "Kanban." [Online]. Available: https://www.atlassian.com/agile/kanban

[23] "Feature driven development." [Online]. Available: http://www.featuredrivendevelopment.com/,

[24] "Dsdm consortium." [Online]. Available: http://www.dsdm.org/

[25] "Lean software institute." [Online]. Available: http://www.leansoftwareinstitute.com/

[26] A. Cockburn, "Software development as a cooperative game," April 2009. [Online]. Available: http://alistair.cockburn.us/Software+development+as+a+cooperative+game

[27] I. E. Certification, "What is agile model." [Online]. Available: http://istqbexamcertification.com/what-is-agile-model-advantages-disadvantages-and-when-to-use-it/

[28] S. J. Schwaber, K., "Scrum guide." [Online]. Available: https://www.scrumguides.org/docs/scrumguide/v1/scrum-guide-us.pdf/

[29] D. N. Card, *The role of measurement in Software Engineering.* Second IEEIBCS Conference, 1988.

[30] C. Institute, "Cmmi the agile way." [Online]. Available: http://cmmiinstitute.com/sites/default/files/resource_asset/CMMI%20The%20Agile%20Way.pdf

[31] "Cmmi maturity levels." [Online]. Available: http://www.tutorialspoint.com/cmmi/cmmi-maturity-levels.htm

[32] "Kpi – key performance indicators." [Online]. Available: https://www.tuv.com/pt/portugal/servicos/gestao_sistemas/qualidade/iso_9001_pt/iso-9001-2015-revisao.html

[33] R. Hughes, *Practical Software Measurement.* McGraw-Hill, April.

[34] S. Nerur, *The intellectual structure of the strategic management field.* John Wiley and Sons Ltd, 2007.

[35] ISO/IEC9126-1, *Software Engineering - Product Qualitty Part 1: Quality Model*, 2001.

[36] P. Kulik, *A pratical approach to software metrics.* IT Professional, 2000.

[37] *Standard for a Software Quality Metrics Methodology.* IEEE Quality Metrics Standard Cornmittee:P1061, 1987.

[38] E. N. Fenton and S. L. Pfleeger, *Software Metrics: A Rigorous and Pracical Approach.* Course Technolgy, 2nd Edition, 1998.

[39] A. Abran, *Software Metrics and Software Metrology.* IEEE Computer Society., September 2010.

[40] J. Kaskinen, *Creating a best-in-class KPI program.* Strategic Finance, 2007.

[41] D. Parmenter, *Key performance indicators: developing, implementing, and using winning kpis.* John Wiley and Sons, Inc., Hoboken, New Jersey, 2010.

[42] Rheinland, "Gestão da qualidade do futuro." [Online]. Available: http://www.pnmsoft.com/resources/bpm-tutorial/key-performance-indicators/

[43] S. H. Kan, *Metrics and models in software quality engineering.* Boston, MA, USA: Addinson-Wesley Longman Publishing Co, Inc., 2000.

[44] M. A. Murray and D. Flaherty, *Can benchmarking give you a competitive edge?* Management Accounting, August.

[45] D. O'Mara, *Benchmarking for profit.* Security Distributing and Marketing, 1999.

[46] A. SHYE, *Code Coverage Testing Using Hardware Performance Monitoring Support.* Department of Electrical and Computer Engineering University of Colorado at Boulder, 2005.

[47] R. OHJA, *Analysis of Key Performance Indicators in Software Development.* TAMPERE UNIVERSITY OF TECHNOLOGY, May 2014.

[48] Z. Antolic, *An Example of Using Key Performance Indicators for Software Development Process Efficiency Evaluation.* Center Ericsson Nikola Tesla, 2008.

[49] M. do Planeamento Orçamento e Gestão, *Indicadores e Métricas para a avaliação de e-Serviços.* Departamento de Governo Eletrônico (DGE) – Secretaria de Logística e Tecnologia da Informação (SLTI) – Ministério do Planejamento, Orçamento e Gestão (MPOG)., Outubro 2007.

[50] "Taskmanagementsoft." [Online]. Available: http://www.taskmanagementsoft.com/products/taskmanagerpro/tutorials/customization-guide/

[51] "Benchmark portal." [Online]. Available: https://www.benchmarkportal.com/calltalk-caramels-tips-on-adherence-to-schedule-adherence/

[52] "Benchmarking metrics software." [Online]. Available: https://www.swtestacademy.com/software-testing-metrics-kpi/

[53] S. McConnel, *CODE COMPLETE 2nd Edition.* Microsoft Press, 2004.

ANEXO A: MEDIÇÕES

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 1	Defect	5	Done	3-Medium			31-Jul-17	3
Sprint 1	Defect	5	Done	2-High			1-Aug-17	3
Sprint 1	Defect	0	Done	4-Low			2-Aug-17	2
Sprint 1	Defect	5	Done	3-Medium			11-Aug-17	3
Sprint 1	Defect	5	Done	4-Low			11-Aug-17	3
Sprint 1	Defect	5	Done	3-Medium			14-Aug-17	3
Sprint 1	Defect	5	Done	4-Low			17-Aug-17	3
Sprint 1	Defect	5	Done	3-Medium			28-Aug-17	3
Sprint 1	Defect	5	Done	2-High			4-Sep-17	5
Sprint 1	User Story	20	Done		21-Jul-17	10:13:40	4-Sep-17	41
Sprint 1	Defect	5	Done	3-Medium			4-Sep-17	5
Sprint 1	Defect	5	Done	3-Medium			5-Sep-17	3
Sprint 1	User Story	10	Done		12-May-17	11:05:23	5-Sep-17	4
Sprint 1	User Story	10	Done		22-May-17	09:31:46	5-Sep-17	4
Sprint 1	User Story	20	Done		29-Mar-17	09:27:39	5-Sep-17	6
Sprint 1	User Story	10	Done		1-Jun-17	13:23:37	5-Sep-17	3
Sprint 1	User Story	10	Done		1-Jun-17	13:25:11	5-Sep-17	3
Sprint 1	User Story	10	Done		1-Jun-17	13:32:12	5-Sep-17	3
Sprint 1	User Story	10	Done		1-Jun-17	13:46:18	5-Sep-17	8
Sprint 1	User Story	20	Done		9-Aug-17	14:32:12	6-Sep-17	5
Sprint 1	User Story	10	Done		1-Jun-17	13:21:21	6-Sep-17	8
Sprint 1	User Story	10	Done		9-Feb-17	08:26:13	6-Sep-17	3
Sprint 1	User Story	10	Done		29-May-17	11:38:24	6-Sep-17	2
Sprint 1	User Story	10	Done		12-May-17	11:12:28	6-Sep-17	2
Sprint 1	User Story	10	Done		9-Feb-17	08:23:49	6-Sep-17	4
Sprint 1	User Story	10	Done		9-Feb-17	08:24:01	6-Sep-17	3
Sprint 1	User Story	10	Done		9-Feb-17	08:24:13	6-Sep-17	3
Sprint 1	User Story	10	Done		1-Jun-17	13:44:19	6-Sep-17	3
Sprint 1	User Story	10	Done		9-Feb-17	08:24:16	6-Sep-17	3
Sprint 1	Defect	5	Done	2-High			6-Sep-17	5
Sprint 1	Defect	5	Done	2-High			7-Sep-17	6
Sprint 1	User Story	15	Done		5-Sep-17	09:47:32	8-Sep-17	3
Sprint 1	Defect	5	Done	2-High			8-Sep-17	3
Sprint 1	Defect	5	Done	4-Low			8-Sep-17	3
Sprint 1	User Story	10	Done		1-Sep-17	15:40:14	8-Sep-17	5
Sprint 1	User Story	10	Done		1-Sep-17	15:45:36	8-Sep-17	9
Sprint 1	User Story	20	Done		20-Jul-17	16:05:06	8-Sep-17	16
Sprint 1	User Story	0	Done		30-Aug-17	12:38:58	11-Sep-17	
Sprint 1	User Story	0	Done		11-Aug-17	14:20:59	11-Sep-17	
Sprint 1	User Story	0	Done		5-Sep-17	10:51:31	11-Sep-17	3
Sprint 1	User Story	30	Done		5-Sep-17	10:56:57	11-Sep-17	6
Sprint 1	Defect	5	Done	3-Medium			12-Sep-17	3
Sprint 1	User Story	50	Done		27-Jul-17	21:28:28	12-Sep-17	
Sprint 1	User Story	20	Done		27-Jul-17	21:32:27	12-Sep-17	

Figura A1: Medição do Sprint 1 referente à entrega 1, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 1	User Story	20	Done		27-Jul-17	21:33:37	12-Sep-17	
Sprint 1	User Story	20	Done		22-May-17	09:20:07	12-Sep-17	7
Sprint 1	User Story	50	Done		27-Jul-17	21:35:05	12-Sep-17	
Sprint 1	User Story	20	Done		27-Jul-17	21:38:39	12-Sep-17	
Sprint 1	User Story	20	Done		29-Mar-17	09:27:43	12-Sep-17	21
Sprint 1	User Story	20	Done		22-May-17	09:17:05	12-Sep-17	8
Sprint 1	User Story	20	Done		29-Mar-17	09:27:41	12-Sep-17	28
Sprint 1	Defect	5	Done	2-High			13-Sep-17	3
Sprint 1	User Story	10	Done		4-Jul-17	09:35:31	14-Sep-17	2
Sprint 1	Defect	5	Done	3-Medium			14-Sep-17	3
Sprint 1	User Story	50	Done		17-Aug-17	09:12:36	14-Sep-17	54
Sprint 1	User Story	20	Done		27-Jul-17	21:29:53	14-Sep-17	
Sprint 1	User Story	20	Done		31-Jul-17	14:42:03	14-Sep-17	18
Sprint 1	User Story	20	Done		29-Mar-17	09:27:45	14-Sep-17	8
Sprint 1	Defect	5	Done	3-Medium			14-Sep-17	4
Sprint 1	Defect	5	Done	2-High			14-Sep-17	3
Sprint 1	User Story	20	Done		8-Feb-17	17:46:16	15-Sep-17	22
Sprint 1	User Story	20	Done		27-Jul-17	21:23:08	15-Sep-17	6
Sprint 1	User Story	20	Done		27-Jul-17	21:37:24	15-Sep-17	
Sprint 1	Defect	5	Done	3-Medium			15-Sep-17	3
Sprint 1	Defect	5	Done	3-Medium			18-Sep-17	3
Sprint 1	Defect	5	Done	3-Medium			18-Sep-17	3
Sprint 1	Defect	5	Done	3-Medium			18-Sep-17	3
Sprint 1	Defect	5	Done	2-High			18-Sep-17	3
Sprint 1	Defect	5	Done	2-High			26-Sep-17	3
Sprint 1	Defect	5	Done	2-High			30-Nov-17	3
Sprint 1	Defect	5	Done	2-High			30-Nov-17	3
Sprint 2	Defect	5	Done	3-Medium			18-Sep-17	3
Sprint 2	Defect	5	Done	4-Low			18-Sep-17	3
Sprint 2	User Story	20	Done		9-Feb-17	08:23:24	18-Sep-17	10
Sprint 2	Defect	5	Done	3-Medium			18-Sep-17	11
Sprint 2	User Story	0	Done		18-Sep-17	17:03:54	19-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			19-Sep-17	4
Sprint 2	Defect	5	Done	3-Medium			20-Sep-17	15
Sprint 2	Defect	5	Done	2-High			21-Sep-17	3
Sprint 2	Defect	5	Done	2-High			21-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			21-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			21-Sep-17	3
Sprint 2	Defect	5	Done	2-High			21-Sep-17	4
Sprint 2	Defect	5	Done	2-High			22-Sep-17	3
Sprint 2	User Story	20	Done		27-Jul-17	21:31:00	22-Sep-17	
Sprint 2	User Story	20	Done		8-Feb-17	17:52:41	25-Sep-17	14
Sprint 2	Defect	5	Done	3-Medium			25-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			26-Sep-17	3

Figura A2: Medição do Sprint 1 e 2 referente à entrega 1, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 2	Defect	5	Done	3-Medium			26-Sep-17	3
Sprint 2	Defect	5	Done	2-High			26-Sep-17	13
Sprint 2	Defect	5	Done	2-High			27-Sep-17	3
Sprint 2	User Story	20	Done		27-Jul-17	21:39:28	27-Sep-17	
Sprint 2	Defect	5	Done	2-High			27-Sep-17	3
Sprint 2	Defect	5	Done	2-High			27-Sep-17	3
Sprint 2	User Story	0	Done		26-Sep-17	12:57:02	28-Sep-17	3
Sprint 2	Defect	5	Done	2-High			28-Sep-17	3
Sprint 2	Defect	5	Done	2-High			28-Sep-17	3
Sprint 2	Defect	5	Done	2-High			28-Sep-17	3
Sprint 2	Defect	5	Done	2-High			28-Sep-17	3
Sprint 2	User Story	0	Done		19-Sep-17	09:35:14	28-Sep-17	
Sprint 2	Defect	5	Done	3-Medium			29-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			29-Sep-17	3
Sprint 2	Defect	5	Done	3-Medium			29-Sep-17	3
Sprint 2	Defect	5	Done	2-High			29-Sep-17	3
Sprint 2	User Story	0	Done		29-Sep-17	14:28:11	29-Sep-17	26
Sprint 2	Defect	5	Done	3-Medium			29-Sep-17	3
Sprint 2	Defect	5	Done	2-High			29-Sep-17	5
Sprint 2	Defect	5	Done	3-Medium			29-Sep-17	3
Sprint 2	Defect	5	Done	2-High			2-Oct-17	5
Sprint 2	User Story	0	Done		29-Sep-17	14:33:51	5-Oct-17	8
Sprint 2	Defect	5	Done	3-Medium			30-Nov-17	3
Sprint 2	Defect	5	Done	2-High			30-Nov-17	3
Sprint 2	Defect	5	Done	2-High			30-Nov-17	3
Sprint 3	Defect	5	Done	3-Medium			22-Sep-17	1
Sprint 3	Defect	5	Done	4-Low			25-Sep-17	1
Sprint 3	Defect	5	Done	2-High			2-Oct-17	3
Sprint 3	Defect	5	Done	2-High			3-Oct-17	3
Sprint 3	Defect	5	Done	2-High			3-Oct-17	4
Sprint 3	Defect	5	Done	1-Critical			3-Oct-17	3
Sprint 3	Defect	5	Done	2-High			3-Oct-17	3
Sprint 3	User Story	20	Done		6-Jun-17	08:55:15	3-Oct-17	9
Sprint 3	User Story	20	Done		6-Jun-17	08:56:15	3-Oct-17	9
Sprint 3	Defect	5	Done	4-Low			4-Oct-17	3
Sprint 3	Defect	5	Done	2-High			4-Oct-17	3
Sprint 3	User Story	10	Done		8-Sep-17	13:37:00	4-Oct-17	2
Sprint 3	User Story	20	Done		18-Sep-17	14:30:32	4-Oct-17	6
Sprint 3	Defect	5	Done	2-High			5-Oct-17	3
Sprint 3	Defect	5	Done	4-Low			6-Oct-17	3
Sprint 3	User Story	10	Done		24-Jul-17	08:59:07	9-Oct-17	33
Sprint 3	Defect	5	Done	3-Medium			10-Oct-17	3
Sprint 3	Defect	5	Done	3-Medium			10-Oct-17	5
Sprint 3	Defect	5	Done	2-High			10-Oct-17	6

Figura A3: Medição do Sprint 2 e 3 referente à entrega 1, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 3	User Story	10	Done		8-Sep-17	13:35:36	10-Oct-17	5
Sprint 3	User Story	10	Done		20-Sep-17	09:06:46	10-Oct-17	2
Sprint 3	User Story	20	Done		3-Oct-17	13:46:21	10-Oct-17	20
Sprint 3	Defect	5	Done	3-Medium			10-Oct-17	5
Sprint 3	User Story	10	Done		5-Sep-17	15:51:14	11-Oct-17	15
Sprint 3	User Story	20	Done		26-May-17	09:53:03	11-Oct-17	14
Sprint 3	Defect	5	Done	2-High			12-Oct-17	3
Sprint 3	Defect	5	Done	2-High			12-Oct-17	3
Sprint 3	User Story	20	Done		3-Jul-17	13:57:09	12-Oct-17	11
Sprint 3	Defect	5	Done	2-High			12-Oct-17	13
Sprint 3	Defect	5	Done	2-High			12-Oct-17	5
Sprint 3	Defect	5	Done	3-Medium			12-Oct-17	3
Sprint 3	Defect	5	Done	2-High			12-Oct-17	3
Sprint 3	Defect	5	Done	2-High			12-Oct-17	3
Sprint 3	User Story	20	Done		27-Jul-17	21:43:03	12-Oct-17	
Sprint 3	Defect	5	Done	2-High			13-Oct-17	3
Sprint 3	Defect	5	Done	4-Low			13-Oct-17	2
Sprint 3	User Story	0	Done		9-Feb-17	08:21:55	13-Oct-17	0
Sprint 3	User Story	0	Done		13-Oct-17	14:38:28	13-Oct-17	12
Sprint 3	Defect	5	Done	3-Medium			13-Oct-17	9
Sprint 3	User Story	50	Done		7-Sep-17	09:54:57	13-Oct-17	6
Sprint 3	User Story	0	Done		13-Oct-17	14:47:40	15-Oct-17	18
Sprint 3	Defect	5	Done	3-Medium			16-Oct-17	4
Sprint 3	User Story	0	Done		13-Oct-17	14:45:31	16-Oct-17	12
Sprint 3	Defect	5	Done	2-High			18-Oct-17	3
Sprint 3	Defect	5	Done	2-High			25-Oct-17	3
Sprint 3	Defect	5	Done	2-High			2-Nov-17	3
Sprint 3	Defect	5	Done	3-Medium			6-Nov-17	2
Sprint 3	User Story	0	Done		18-May-17	10:33:07	29-Nov-17	2
Sprint 3	Defect	5	Done	3-Medium			30-Nov-17	3
Sprint 3	Defect	5	Done	4-Low			30-Nov-17	3
Sprint 3	Defect	5	Done	3-Medium			30-Nov-17	4
Sprint 3	Defect	5	Done	3-Medium			30-Nov-17	8
Sprint 3	Defect	5	Done	3-Medium			30-Nov-17	4
Sprint 4	Defect	5	Done	2-High			18-Oct-17	12
Sprint 4	User Story	20	Done		1-Sep-17	15:34:15	18-Oct-17	33
Sprint 4	Defect	5	Done	3-Medium			18-Oct-17	3
Sprint 4	User Story	10	Done		10-May-17	12:40:36	19-Oct-17	3
Sprint 4	User Story	20	Done		27-Jul-17	21:36:27	19-Oct-17	8
Sprint 4	User Story	20	Done		29-Sep-17	09:19:36	20-Oct-17	40
Sprint 4	User Story	10	Done		4-Oct-17	10:20:00	23-Oct-17	5
Sprint 4	User Story	20	Done		16-Oct-17	14:49:40	23-Oct-17	21
Sprint 4	User Story	10	Done		9-Feb-17	08:26:53	23-Oct-17	10
Sprint 4	User Story	20	Done		9-Feb-17	08:23:45	24-Oct-17	18

Figura A4: Medição do Sprint 3 e 4 referente à entrega 1, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 4	User Story	20	Done		6-Oct-17	15:30:51	26-Oct-17	6
Sprint 4	Defect	5	Done	2-High			27-Oct-17	9
Sprint 4	Defect	5	Done	2-High			27-Oct-17	3
Sprint 4	User Story	20	Done		12-Jul-17	17:03:28	27-Oct-17	24
Sprint 4	Defect	5	Done	2-High			27-Oct-17	2
Sprint 4	User Story	20	Done		5-Oct-17	14:47:45	30-Oct-17	13
Sprint 4	Defect	5	Done	2-High			30-Oct-17	3
Sprint 4	Defect	5	Done	3-Medium			30-Oct-17	3
Sprint 4	Defect	5	Done	3-Medium			30-Oct-17	3
Sprint 4	User Story	20	Done		6-Oct-17	15:18:47	30-Oct-17	13
Sprint 4	User Story	20	Done		7-Aug-17	12:44:27	30-Oct-17	21
Sprint 4	Defect	5	Done	2-High			30-Oct-17	11
Sprint 4	Defect	5	Done	3-Medium			1-Nov-17	3
Sprint 4	Defect	5	Done	3-Medium			2-Nov-17	10
Sprint 4	Defect	5	Done	2-High			3-Nov-17	5
Sprint 4	Defect	5	Done	3-Medium			3-Nov-17	3
Sprint 4	User Story	20	Done		18-May-17	08:12:55	3-Nov-17	17
Sprint 4	Defect	5	Done	3-Medium			3-Nov-17	5
Sprint 4	Defect	5	Done	3-Medium			3-Nov-17	2
Sprint 4	User Story	5	Done		30-Oct-17	17:19:51	3-Nov-17	6
Sprint 4	User Story	10	Done		26-Oct-17	11:31:39	3-Nov-17	7
Sprint 4	User Story	4	Done		31-Oct-17	15:22:16	3-Nov-17	6
Sprint 4	User Story	0	Done		13-Oct-17	14:48:20	5-Nov-17	22
Sprint 4	User Story	0	Done		13-Oct-17	14:52:27	5-Nov-17	33
Sprint 4	User Story	0	Done		5-Nov-17	22:46:16	5-Nov-17	26
Sprint 4	User Story	0	Done		5-Nov-17	22:46:17	5-Nov-17	12
Sprint 4	User Story	0	Done		5-Nov-17	22:54:34	5-Nov-17	12
Sprint 4	Defect	5	Done	3-Medium			6-Nov-17	3
Sprint 4	User Story	10	Done		4-Jul-17	16:03:20	6-Nov-17	7
Sprint 4	User Story	0	Done		5-Nov-17	22:57:30	6-Nov-17	29
Sprint 4	Defect	5	Done	3-Medium			15-Nov-17	3
Sprint 4	Defect	5	Done	3-Medium			5-Dec-17	3
Sprint 4	Defect	5	Done	3-Medium			5-Dec-17	16
Sprint 5	Defect	5	Done	3-Medium			7-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			8-Nov-17	7
Sprint 5	Defect	5	Done	3-Medium			8-Nov-17	3
Sprint 5	User Story	20	Done		17-Oct-17	07:42:28	8-Nov-17	32
Sprint 5	User Story	10	Done		6-Nov-17	07:59:49	9-Nov-17	14
Sprint 5	User Story	50	Done		27-Jul-17	21:20:00	9-Nov-17	13
Sprint 5	Defect	5	Done	2-High			9-Nov-17	3
Sprint 5	User Story	20	Done		3-Nov-17	08:44:29	13-Nov-17	28
Sprint 5	Defect	5	Done	3-Medium			13-Nov-17	4
Sprint 5	Defect	5	Done	3-Medium			13-Nov-17	7
Sprint 5	User Story	20	Done		30-Oct-17	16:02:03	13-Nov-17	22

Figura A5: Medição do Sprint 4 e 5 referente à entrega 1, do projeto I da KPI Consulting.

Sprint 5	User Story	0	Done		2-Nov-17	15:23:56	13-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			13-Nov-17	2
Sprint 5	Defect	5	Done	3-Medium			13-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			13-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			14-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			15-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			15-Nov-17	16
Sprint 5	User Story	50	Done		8-Feb-17	17:52:49	15-Nov-17	32
Sprint 5	User Story	50	Done		9-Feb-17	08:26:40	15-Nov-17	29
Sprint 5	Defect	5	Done	2-High			16-Nov-17	4
Sprint 5	User Story	50	Done		13-Mar-17	11:54:56	16-Nov-17	50
Sprint 5	Defect	5	Done	2-High			16-Nov-17	9
Sprint 5	Defect	5	Done	3-Medium			16-Nov-17	3
Sprint 5	User Story		Done		13-Nov-17	08:20:20	16-Nov-17	16
Sprint 5	User Story	0	Done		15-Nov-17	11:36:06	16-Nov-17	5
Sprint 5	Defect	5	Done	3-Medium			17-Nov-17	2
Sprint 5	Defect	5	Done	3-Medium			17-Nov-17	3
Sprint 5	Defect	5	Done	2-High			17-Nov-17	24
Sprint 5	User Story	0	Done		15-Nov-17	11:40:03	17-Nov-17	2
Sprint 5	User Story	20	Done		26-Oct-17	12:04:23	17-Nov-17	31
Sprint 5	User Story	0	Done		19-Nov-17	22:21:33	20-Nov-17	28
Sprint 5	Defect	5	Done	2-High			20-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			20-Nov-17	5
Sprint 5	Defect	5	Done	3-Medium			20-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			30-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			30-Nov-17	3
Sprint 5	Defect	5	Done	3-Medium			30-Nov-17	10

Figura A6: Medição do Sprint 5 referente à entrega 1, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 1	Defect	5	Done	4-Low			20-Nov-17	
Sprint 1	User Story	10	Done		27-Sep-17	10:07:22	30-Nov-17	8
Sprint 1	Defect	5	Done	4-Low			5-Dec-17	5
Sprint 1	Defect	5	Done	4-Low			20-Nov-17	
Sprint 1	Defect	5	Done	2-High			20-Nov-17	4
Sprint 1	Defect	5	Done	2-High			23-Nov-17	2
Sprint 1	Defect	5	Done	2-High			19-Dec-17	
Sprint 1	Defect	5	Done	2-High			21-Nov-17	3
Sprint 1	Defect	5	Done	3-Medium			22-Nov-17	3
Sprint 1	Defect	5	Done	1-Critical			23-Nov-17	3
Sprint 1	Defect	5	Done	2-High			30-Nov-17	
Sprint 1	Defect	5	Done	3-Medium			5-Dec-17	4
Sprint 1	Defect	5	Done	3-Medium			23-Nov-17	2
Sprint 1	Defect	5	Done	2-High			22-Nov-17	2
Sprint 1	Defect	5	Done	4-Low			17-Nov-17	3
Sprint 1	Defect	5	Done	4-Low			1-Dec-17	4
Sprint 1	Defect	5	Done	3-Medium			7-Dec-17	4
Sprint 1	Defect	5	Done	3-Medium			27-Nov-17	3
Sprint 1	Defect	5	Done	3-Medium			30-Nov-17	2
Sprint 1	Defect	5	Done	2-High			30-Nov-17	5
Sprint 1	Defect	5	Done	3-Medium			20-Nov-17	2
Sprint 1	Defect	5	Done	3-Medium			30-Nov-17	3
Sprint 1	Defect	5	Done	3-Medium			17-Nov-17	8
Sprint 1	Defect	5	Done	3-Medium			5-Dec-17	9
Sprint 1	Defect	5	Done	3-Medium			30-Nov-17	9
Sprint 1	Defect	5	Done	3-Medium			3-Jan-18	8
Sprint 1	Defect	5	Done	3-Medium			24-Nov-17	10
Sprint 1	Defect	5	Done	3-Medium			24-Nov-17	3
Sprint 1	Defect	5	Done	3-Medium			20-Nov-17	1
Sprint 1	Defect	5	Done	3-Medium			23-Nov-17	5
Sprint 1	Defect	5	Done	2-High			24-Nov-17	5
Sprint 1	Defect	5	Done	3-Medium			23-Nov-17	2
Sprint 1	Defect	5	Done	4-Low			21-Nov-17	5
Sprint 1	Defect	5	Done	4-Low			7-Dec-17	4
Sprint 1	Defect	5	Done	4-Low			30-Nov-17	5
Sprint 1	Defect	5	Done	3-Medium			29-Nov-17	11
Sprint 1	Defect	5	Done	2-High			9-Nov-17	4
Sprint 1	Defect	5	Done	2-High			29-Nov-17	18
Sprint 1	Defect	5	Done	3-Medium			30-Nov-17	5
Sprint 1	Defect	5	Done	2-High			29-Nov-17	11
Sprint 1	Defect	5	Done	2-High			30-Nov-17	3
Sprint 1	Defect	5	Done	3-Medium			22-Nov-17	2
Sprint 1	Defect	5	Done	3-Medium			29-Nov-17	2
Sprint 1	Defect	5	Done	2-High			6-Dec-17	3

Figura A7: Medição do Sprint 1 referente à entrega 2, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 1	Defect	5	Done	3-Medium			9-Nov-17	3
Sprint 1	User Story	0	Done		20-Nov-17	07:58:58	24-Nov-17	9
Sprint 1	User Story	20	Done		20-Nov-17	15:04:58	4-Dec-17	29
Sprint 1	User Story	50	Done		28-Aug-17	14:46:19	30-Nov-17	21
Sprint 1	User Story	10	Done		1-Jun-17	13:58:53	23-Nov-17	10
Sprint 1	User Story	10	Done		1-Jun-17	14:00:48	24-Nov-17	11
Sprint 1	User Story	0	Done		4-Jul-17	09:18:09	2-Nov-17	
Sprint 1	User Story	10	Done		6-Oct-17	15:44:41	17-Nov-17	10
Sprint 1	User Story	50	Done		6-Oct-17	15:25:41	4-Dec-17	13
Sprint 1	User Story	10	Done		16-Nov-17	12:03:16	20-Nov-17	8
Sprint 1	User Story	10	Done		14-Jul-17	16:45:38	30-Nov-17	8
Sprint 1	Defect	5	Done	4-Low			28-Nov-17	3
Sprint 2	Defect	5	Done	3-Medium			5-Dec-17	1
Sprint 2	Defect	5	Done	2-High			3-Jan-18	14
Sprint 2	Defect	5	Done	2-High			13-Dec-17	9
Sprint 2	Defect	5	Done	2-High			5-Dec-17	3
Sprint 2	Defect	5	Done	2-High			14-Dec-17	2
Sprint 2	Defect	5	Done	2-High			29-Nov-17	9
Sprint 2	Defect	5	Done	3-Medium			9-Nov-17	4
Sprint 2	Defect	5	Done	3-Medium			21-Nov-17	3
Sprint 2	Defect	5	Done	4-Low			29-Nov-17	6
Sprint 2	Defect	5	Done	3-Medium			17-Nov-17	3
Sprint 2	Defect	5	Done	4-Low			6-Dec-17	5
Sprint 2	Defect	5	Done	3-Medium			11-Jan-18	3
Sprint 2	Defect	5	Done	2-High			10-Dec-17	5
Sprint 2	Defect	5	Done	3-Medium			22-Nov-17	5
Sprint 2	Defect	5	Done	3-Medium			30-Nov-17	1
Sprint 2	Defect	5	Done	2-High			13-Dec-17	5
Sprint 2	Defect	5	Done	2-High			5-Dec-17	
Sprint 2	Defect		Done	2-High			16-Aug-17	3
Sprint 2	Defect	5	Done	3-Medium			23-Nov-17	3
Sprint 2	Defect	5	Done	3-Medium			21-Nov-17	3
Sprint 2	Defect	5	Done	2-High			13-Dec-17	4
Sprint 2	Defect	5	Done	4-Low			7-Dec-17	4
Sprint 2	Defect	5	Done	2-High			13-Dec-17	12
Sprint 2	Defect	5	Done	4-Low			5-Dec-17	14
Sprint 2	Defect	5	Done	3-Medium			21-Nov-17	3
Sprint 2	Defect	5	Done	4-Low			30-Nov-17	4
Sprint 2	Defect	5	Done	3-Medium			30-Nov-17	5
Sprint 2	Defect	5	Done	3-Medium			21-Nov-17	3
Sprint 2	Defect	5	Done	3-Medium			23-Nov-17	2
Sprint 2	Defect	5	Done	3-Medium			13-Nov-17	4

Figura A8: Medição do Sprint 1 e 2 referente à entrega 2, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 2	Defect	5	Done	3-Medium			14-Nov-17	5
Sprint 2	Defect	5	Done	4-Low			29-Nov-17	3
Sprint 2	Defect	5	Done	3-Medium			7-Dec-17	3
Sprint 2	Defect	5	Done	2-High			15-Dec-17	12
Sprint 2	Defect	5	Done	3-Medium			28-Nov-17	2
Sprint 2	Defect	5	Done	3-Medium			12-Dec-17	3
Sprint 2	Defect	5	Done	4-Low			7-Dec-17	3
Sprint 2	Defect	5	Done	2-High			14-Dec-17	3
Sprint 2	Defect	5	Done	3-Medium			15-Dec-17	6
Sprint 2	Defect	5	Done	3-Medium			15-Dec-17	7
Sprint 2	Defect	5	Done	2-High			15-Dec-17	5
Sprint 2	Defect	5	Done	2-High			13-Dec-17	1
Sprint 2	User Story	20	Done		8-Feb-17	17:59:54	13-Dec-17	10
Sprint 2	User Story	50	Done		15-Mar-17	09:18:01	19-Dec-17	32
Sprint 2	User Story	20	Done		24-Mar-17	15:01:38	15-Dec-17	12
Sprint 2	User Story	50	Done		15-Nov-17	17:04:46	14-Dec-17	60
Sprint 2	User Story	20	Done		6-Sep-17	11:24:56	19-Dec-17	36
Sprint 2	User Story	20	Done		21-Nov-17	10:33:54	14-Dec-17	14
Sprint 2	User Story	50	Done		29-Nov-17	10:04:45	12-Dec-17	12
Sprint 2	User Story	50	Done		26-May-17	08:46:34	12-Dec-17	30
Sprint 2	User Story	10	Done		27-Sep-17	14:06:59	4-Dec-17	9
Sprint 2	User Story	50	Done		5-Dec-17	09:42:52	19-Dec-17	52
Sprint 2	User Story	0	Done		6-Dec-17	12:05:25	11-Dec-17	4
Sprint 2	User Story	50	Done		18-Oct-17	12:56:22	7-Dec-17	26
Sprint 2	User Story	50	Done		29-Nov-17	10:01:57	7-Dec-17	8
Sprint 2	User Story	20	Done		27-Sep-17	12:58:11	13-Dec-17	8
Sprint 2	User Story	50	Done		14-Jun-17	07:57:54	15-Dec-17	24
Sprint 2	User Story	20	Done		21-Nov-17	10:37:23	11-Dec-17	10
Sprint 2	User Story	0	Done		28-Nov-17	17:22:36	19-Dec-17	20
Sprint 2	User Story	0	Done		14-Dec-17	11:10:25	15-Dec-17	9
Sprint 3	Defect	5	Done	3-Medium			20-Dec-17	3
Sprint 3	Defect	5	Done	3-Medium			21-Dec-17	3
Sprint 3	User Story	0	Done		12-Jan-18	14:44:09	12-Jan-18	10
Sprint 3	User Story	0	Done		4-Jan-18	19:12:52	15-Jan-18	31
Sprint 3	User Story	10	Done		8-Jun-17	13:00:03	27-Dec-17	16
Sprint 3	User Story	0	Done		26-Dec-17	14:39:49	27-Dec-17	15
Sprint 3	User Story	0	Done		18-Dec-17	17:01:44	26-Dec-17	18
Sprint 3	User Story	50	Done		11-Aug-17	17:40:39	15-Jan-18	30
Sprint 3	User Story	20	Done		3-Oct-17	15:19:11	8-Jan-18	8
Sprint 3	User Story	20	Done		16-Aug-17	15:43:30	18-Dec-17	13
Sprint 3	User Story	20	Done		14-Jul-17	16:47:40	5-Jan-18	23
Sprint 3	User Story	50	Done		18-Oct-17	13:18:34	20-Dec-17	34

Figura A9: Medição do Sprint 2 e 3 referente à entrega 2, do projeto I da KPI Consulting.

Sprint	Type	Story Points	Status	Severity	Creation Date	Creation Time	Done on date	Invested (Hours)
Sprint 3	User Story	20	Done		13-Nov-17	11:57:42	22-Dec-17	3
Sprint 3	User Story	5	Done		24-Apr-17	08:00:14	21-Dec-17	3
Sprint 3	User Story	10	Done		6-Dec-17	07:31:49	22-Dec-17	3
Sprint 3	User Story	10	Done		6-Dec-17	07:29:16	20-Dec-17	3
Sprint 3	User Story	0	Done		16-Oct-17	11:49:35	29-Dec-17	60
Sprint 3	User Story	50	Done		20-Nov-17	15:05:09	20-Dec-17	48
Sprint 3	Defect	5	Done	4-Low			29-Dec-17	3
Sprint 3	Defect	5	Done	3-Medium			5-Jan-18	2
Sprint 3	Defect	5	Done	3-Medium			29-Dec-17	3
Sprint 3	Defect	5	Done	3-Medium			5-Jan-18	2
Sprint 3	Defect	5	Done	4-Low			29-Dec-17	3
Sprint 3	Defect	10	Done	3-Medium			12-Jan-18	45
Sprint 3	Defect	5	Done	3-Medium			18-Jan-18	13
Sprint 3	Defect	5	Done	3-Medium			19-Dec-17	2
Sprint 3	Defect	5	Done	3-Medium			20-Dec-17	3
Sprint 3	Defect	5	Done	2-High			27-Dec-17	18
Sprint 3	Defect	5	Done	3-Medium			22-Dec-17	3
Sprint 3	Defect	5	Done	3-Medium			11-Jan-18	6
Sprint 3	Defect	5	Done	2-High			21-Dec-17	3
Sprint 3	Defect	5	Done	3-Medium			29-Dec-17	2
Sprint 3	Defect	5	Done	2-High			28-Dec-17	3
Sprint 3	Defect	5	Done	2-High			29-Dec-17	4
Sprint 3	Defect	5	Done	2-High			29-Dec-17	3
Sprint 3	Defect	5	Done	2-High			3-Jan-18	3
Sprint 3	Defect	5	Done	2-High			5-Jan-18	3
Sprint 3	Defect	5	Done	3-Medium			28-Dec-17	8
Sprint 3	Defect	5	Done	3-Medium			27-Dec-17	7
Sprint 3	Defect	5	Done	2-High			27-Dec-17	3
Sprint 3	Defect	5	Done	2-High			5-Jan-18	7
Sprint 3	Defect	5	Done	2-High			5-Jan-18	2
Sprint 3	Defect	5	Done	3-Medium			11-Jan-18	2
Sprint 3	User Story	5	Done		8-Feb-17	17:46:29	22-Dec-17	37
Sprint 3	User Story	10	Done		8-Feb-17	17:57:43	20-Dec-17	7
Sprint 3	User Story	20	Done		13-Dec-17	11:30:15	21-Dec-17	15
Sprint 4	User Story	0	Done		19-Jan-18	09:25:30	19-Jan-18	8
Sprint 4	User Story	0	Done		12-Jan-18	14:55:05	17-Jan-18	10
Sprint 4	Defect	5	Done	3-Medium			16-Jan-18	4
Sprint 4	Defect	5	Done	2-High			22-Jan-18	10
Sprint 4	Defect	5	Done	3-Medium			17-Jan-18	3
Sprint 4	Defect	5	Done	3-Medium			9-Jan-18	2
Sprint 4	User Story	10	Done		10-Jan-18	07:46:45	19-Jan-18	6
Sprint 4	User Story	20	Done		24-Apr-17	08:54:23	22-Jan-18	13

Figura A10: Medição do Sprint 3 e 4 referente à entrega 2, do projeto I da KPI Consulting.

ANEXO B: DATAS DE ENTREGA

Refresh

Date	Application	Release	Build	Deploy level	Transport release		Status	Count:145
				[ES JBoss6 Applications] Produktions Stufe	Admin	Quality		
16.12.2017 00:30	KPI CONSULTING	13.XX	PROJECT I	Produktions Stufe	Z000SWSC approved	Z001PFMR approved	Closed	
30.11.2017 01:00	KPI CONSULTING	14.1.X	PROJECT I	Produktions Stufe	Z002385E approved	Z0000EVD approved	Closed	
25.11.2017 00:30	KPI CONSULTING	14.X	PROJECT I	Produktions Stufe	Z0007NAA approved	Z0000EVD approved	Closed	
30.09.2017 04:00	KPI CONSULTING	13.X_bugfix2	PROJECT I	Produktions Stufe	Z0007NAA approved	Z0000EVD approved	Closed	
28.09.2017 00:30	KPI CONSULTING	13.X_bugfix	PROJECT I	Produktions Stufe	Z0007NAA approved	Z001PFMR approved	Closed	
16.09.2017 00:30	KPI CONSULTING	13.X	PROJECT I	Produktions Stufe	Z0007NAA approved	Z0000EVD approved	Closed	
12.08.2017 00:30	KPI CONSULTING	R12_X_new	PROJECT I	Produktions Stufe	Z0007NAA approved	Z0000EVD approved	Closed	
01.08.2017 17:30	KPI CONSULTING	R12_X_bugfix	PROJECT I	Produktions Stufe	Z003DZTK approved	Z0000EVD approved	Closed	
06.07.2017 00:30	KPI CONSULTING	R12_X	PROJECT I	Produktions Stufe	Z002365E approved	Z0000EVD approved	Closed	
14.06.2017 02:00	KPI CONSULTING	2017.Q1	PROJECT I	Produktions Stufe	Z000DUSS approved	Z0000EVD approved	Closed	
13.06.2017 00:30	KPI CONSULTING	2017.Q1	PROJECT I	Produktions Stufe	Z000DUSS approved	Z0000EVD approved	Error	
10.06.2017 00:30	KPI CONSULTING	2017.Q1	PROJECT I	Produktions Stufe	Z002365E approved	Z0000EVD approved	Error	
06.05.2017 01:30	KPI CONSULTING	2017.Q1	PROJECT I	Produktions Stufe	Z0007NAA approved	Z0000EVD approved	Error	

<< < 1 2 > >>

Figura B1: Datas de entrega do projeto I da KPI Consulting.